53-1003858-02
12 August 2015

Brocade Virtual Traffic Manager: User's Guide

Supporting 10.1

BROCADE

Copyright © 2015 Brocade Communications Systems, Inc. All Rights Reserved.

ADX, Brocade, Brocade Assurance, the B-wing symbol, DCX, Fabric OS, HyperEdge, ICX, MLX, MyBrocade, OpenScript, The Effortless Network, VCS, VDX, Vplane, and Vyatta are registered trademarks, and Fabric Vision and vADX are trademarks of Brocade Communications Systems, Inc., in the United States and/or in other countries. Other brands, products, or service names mentioned may be trademarks of others.

Notice: This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any equipment, equipment feature, or service offered or to be offered by Brocade. Brocade reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact a Brocade sales office for information on feature and product availability. Export of technical data contained in this document may require an export license from the United States government.

.The authors and Brocade Communications Systems, Inc. assume no liability or responsibility to any person or entity with respect to the accuracy of this document or any loss, cost, liability, or damages arising from the information contained herein or the computer programs that accompany it.

The product described by this document may contain “open source” software covered by the GNU General Public License or other open source license agreements. To find out which open source software is included in Brocade products, view the licensing terms applicable to the open source software, and obtain a copy of the programming source code, please visit

http://www.brocade.com/en/support/support-tools/oscd.html.

Brocade Communications Systems, Incorporated

Corporate and Latin American Headquarters
Brocade Communications Systems, Inc.
130 Holger Way
San Jose, CA 95134
Tel: 1-408-333-8000
Fax: 1-408-333-8101
E-mail: info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems China HK, Ltd.
No. 1 Guanghua Road
Chao Yang District
Units 2718 and 2818
Beijing 100020, China
Tel: +8610 6588 8888
Fax: +8610 6588 9999
E-mail: china-info@brocade.com

European Headquarters
Brocade Communications Switzerland Sàrl
Centre Swissair
Tour B - 4ème étage
29, Route de l'Aéroport
Case Postale 105
CH-1215 Genève 15
Switzerland
Tel: +41 22 799 5640
Fax: +41 22 799 5641
E-mail: emea-info@brocade.com

Asia-Pacific Headquarters
Brocade Communications Systems Co., Ltd. (Shenzhen WFOE)
Citic Plaza
No. 233 Tian He Road North
Unit 1308 – 13th Floor
Guangzhou, China
Tel: +8620 3891 2000
Fax: +8620 3891 2111
E-mail: china-info@brocade.com

Contents

Preface 18

About This Guide 18
Audience 18
Document Conventions 18
Documentation and Release Notes 19
Traffic Manager Documentation 20
Traffic Manager Online Help 20
Traffic Manager Information Online 20
Getting Technical Help or Reporting Errors 20
Web Access 21
E-mail and Telephone Access 21

CHAPTER 1 Traffic Manager Overview 22

Introducing the Traffic Manager 22
Typical Deployment 23
Traffic Manager Product Variants 23
Developer Mode 24
Supported Platforms 24
Supported Cluster Combinations 25
Chapter Outline 26

CHAPTER 2 Network Layouts 31

Essentials of Network Configuration 31
Dedicated Management Network 32
Sizing Your Cluster 33
Front-End Servers 33
IP Transparency 35
Routing Configuration 36
Local Routing Problems 37
IP Transparency and Traffic Manager Clusters 37
Traffic IP Addresses and Traffic IP Groups 37
Traffic IP Address Modes 38
Example Configurations 39
Using IP Transparency with a Cluster 42
Route Health Injection and the Network 45
Introduction to IPv6 49

Main Features of IPv6 in the Traffic Manager 49
Technical Restrictions 50

CHAPTER 3 Initial Configuration 52

Architecture Concepts 52
Managing Your First Service 54
Using the Wizard to Create a Virtual Server and Pool 54
Creating a Pool and Virtual Server Manually 55
Creating a Cluster 56
Joining a Cluster 57
Joining Clusters with Traffic IP Groups 60

CHAPTER 4 Virtual Servers 61

Applying Rules 63
SSL Decryption 64
Service Protection Classes 65
Bandwidth Management Classes 66
Service Level Monitoring Classes 66
HTTP Content Caching 67
Aptimizer Settings 67
HTTP Content Compression 68
Controlling Content Compression 68
Connection Analytics 69
Using Connection Analytics 70
Request Logging 70
Request Logging to a File 70
Remote Request Logging 71
Controlling Request Logging 72
Connection Management 72
Handling Errors 76
Memory Limits for Connections 78

CHAPTER 5 Pools 80

Load Balancing 80
Locality Aware Request Distribution (LARD) 82
Session Persistence 83
Bandwidth Management 84
Health Monitoring 84
SSL Encryption 85
Connection Management 85
Pool Connection Limiting 86
Introduction 86

Pool Connection Limits 86
Clustered Connection Limiting 86
Connection Queuing 87
Considerations 87
Enabling Pool Connection Limiting 88
Tracking Connection Limits 88
Testing Connection Limits 89
Back-End Fault Tolerance 89
Draining and Disabling Nodes 91
Using the Drain a Node and Disable a Node Wizards 92
Autoscaling 93
Introduction 93
How It Works 93
Configuration 94
DNS-Derived Autoscaling 103

CHAPTER 6 Traffic IP Groups and Fault Tolerance 105

Fault Tolerance 105
Traffic IP Addresses and Traffic IP Groups 105
Distributing Traffic Within a Traffic IP Group 106
Choosing Traffic IP Addresses 106
Creating a Traffic IP Group 107
Traffic Distribution 107
Passive Machines 109
Disabling a Traffic IP Group 109
Interface-to-Subnet Mapping (Traffic IP Networks) 109
Configuring Fault-Tolerance 110
Fault Tolerance Configuration Settings 110
Understanding Traffic Manager Fault Tolerance Checks 113
Health Broadcasts 113
Determining the Health of a Cluster 114
Failover 114
Traffic IP Address Transfer (Single-Hosted Mode) 114
Traffic IP Address Transfer (Multi-Hosted Mode) 115
Traffic IP Address Transfer (RHI Mode) 115
Recovering from Failure 115
Debugging and Monitoring Fault Tolerance Activity 116
Configuring BGP Connectivity 116
Configuring BGP Router IDs 117
Managing BGP Neighbors 117
Configuring OSPFv2 Connectivity 118
Configuring OSPFv2 IP Addresses 119

Configuring Neighborhood Monitoring 120

CHAPTER 7 Key Features in the Traffic Manager Administration Interface .. 121

The Home Page 121
Services > Configuration Summary 122
Catalogs 122
License Management 123
Adding and Removing License Keys 124
System > Global Settings 125
System > Backups 126
System > Backups > Partial Backups 128
Activity Monitoring 132
Activity > Current Activity 133
Activity > Historical Activity 137
Activity > Map 138
Activity > Connections 138
Activity > Draining Nodes 140
Activity > View Logs 141
Cloud Credentials 141
IAM Roles in Amazon EC2 Credentials 143
The Web Application Firewall 143
Overview 144
Enabling the Application Firewall 144
Application Firewall Features in the Traffic Manager Admin UI 144
The System > Application Firewall Page 146
The Enforcer and Decider 148
The Enforcer Rule 148
User Management 149
Updating Your Software 150

CHAPTER 8 TrafficScript Rules 151

Overview 151
TrafficScript Example 152
TrafficScript Documentation 153
Applications of Rules on the Traffic Manager 154
Using a Rule on the Traffic Manager 154
Creating a Rule in the Catalog 155
Uploading a Rule to the Catalog 158
Applying a Rule to a Virtual Server 159
Example Rules 160
Routing by Content Type 160
Restricting Access Based on Time of Day 160

Customer Prioritization....161
Managing Levels of Service....161
Routing Based on XML Traffic....162

CHAPTER 9 TrafficScript Authentication Support....165

Overview....165
Configuring Authenticators....165
Configuring the TrafficScript Rule....169
Configuring the Virtual Server....170

CHAPTER 10 Java Extensions....171

Introduction to Java....171
Invoking a Java Extension....171
Configuring the Traffic Manager to Use Java....172
Requirements....172
Compiling a Java Extension....172
Loading Java Extensions onto the Traffic Manager....173
Configuring the Traffic Manager's Java Extension Runner....173

CHAPTER 11 Protocol Support....175

Basic TCP Protocols....175
Server-First Protocols....175
Client-First Protocols....176
Server-First with "Server Banner"....177
Generic Streaming Protocols....178
HTTP....178
SSL....179
Protecting the SSL Handshake....180
SSL Connection Renegotiation Protection....180
SMTP (Simple Mail Transport Protocol)....182
FTP....183
FTP Source Ports....184
SSL-Wrapped FTP (FTPS)....185
Use Cases for SSL-Wrapped FTP....186
Real-Time Streaming Protocol....187
Setting Up an RTSP Service....188
Session Initiation Protocol....189
Features of SIP....189
The Traffic Manager and the SIP Protocol....190
Configuring the Proxy Servers to Support Traffic Management....191
Setting Up a SIP Service on the Traffic Manager....192
SIP Operation Modes on the Traffic Manager....192

Additional SIP Settings 194
Communicating with UDP-Based SIP Servers 196

CHAPTER 12 Session Persistence 197

What Is Session Persistence? 197
Configuring Session Persistence 198
Enabling Session Persistence 198
Selecting a Persistence Method 199
Resolving Session Persistence Maps to Nodes 204
Node Failure Options 204
Draining Connections 205
Sizing the Session Persistence Caches 206
Using Session Persistence with Multi-Hosted Traffic IP Addresses 207
Session Persistence with UDP protocols 208
Examples 208
Universal PHP Persistence 208

CHAPTER 13 SSL Encryption 210

Overview of SSL 210
Server Authentication 210
Client Authentication 211
Encrypted Data Transfer 211
SSL Features in the Traffic Manager Traffic Manager 212
Decryption and Encryption 212
SSL Certificates Catalog 212
SSL Decryption Wizard 212
Configuring SSL Certificates 213
Creating a New Self-Signed SSL Certificate 213
Managing Certificate Data 215
Creating a Certificate Signing Request 215
Importing a New SSL Certificate 216
Working with Intermediate Certificates 217
Managing Certificate Authority Certificates and CRL Files 218
SSL Decryption 219
Setting Up SSL Decryption 219
Using Multiple SSL Certificates 220
Configuring Ciphers and TLS Versions 222
Client Certificates 223
Configuring OCSP 224
SSL Session ID Cache 227
OCSP Stapling Cache 228
SSL Encryption 228

Preserving IP Addresses with SSL Forwarding 229
Use of SSL Cryptographic Devices....230
Configuring the Traffic Manager to Use an SSL Device 232
Verifying Correct Operation of SSL Devices....234
Using the Connect to Microsoft Azure Key Vault Wizard 235
Using the Connect to NetHSM Wizard 237
Identifying Keys and Certificates Stored on a Secure Device 240

CHAPTER 14 Health Monitoring 241

Which Nodes Are Monitored?....241
Using Nodes in Multiple Pools 241
Passive Health Monitoring....242
Retrying Failed Requests 243
Node Failures....244
Enabling and Disabling Passive Monitoring 244
Overview of Health Monitors....244
The Monitors Catalog....245
Built-in Health Monitors....246
Custom Health Monitors 248
Per-Node and Pool-Wide Monitors....249
Using Health Monitors 249
Applying a Monitor to a Pool....249
External Program Monitors....250
Uploading Monitors to the Traffic Manager....251
Writing Monitors in Perl....251

CHAPTER 15 Service Protection 253

Classes of Risk....253
Denial of Service (DoS)....253
Web Worms and Viruses 253
Distributed Denial of Service Attacks (DDoS)....253
Malformed HTTP Attacks....254
Firewalls and Other Security Measures....254
Protection Features....254
Network Access Restrictions....254
Connection Limiting....255
Malformed HTTP Filtering....255
Rule-Based Protection 255
Enabling Service Protection....255
Adding a Service Protection Class 256
Basic Settings 256
Simultaneous Connections 256

Connection Rate 257
Access Restrictions 258
HTTP-Specific Settings 258
Service Protection Rule 259
Applying a Service Protection Class to a Virtual Server 259
Service Protection Performance 260

CHAPTER 16 Bandwidth Management 261

What Is Bandwidth Management? 261
Configuring Bandwidth Management 262
Adding a Bandwidth Class to the Catalog 262
Assigning a Bandwidth Class to a Virtual Server 263
Assigning a Bandwidth Class to a Pool 263
Using TrafficScript to Select a Bandwidth Class 264

CHAPTER 17 Request Rate Shaping 265

What Is Request Rate Shaping? 265
Configuring a Request Rate Shaping Class (Rate Class) 266
Adding a Rate Class to the Catalog 266
Using a Rate Class 266
The Rate Queue 267
Selective Rate Shaping 268
More Fine-Grained Rate Shaping 268
Rate-Shaping Web Spiders 269
Graphing Request Rate Shaping 270

CHAPTER 18 Service Level Monitoring 271

Introducing Service Level Monitoring 271
Configuring a Service Level Monitoring Class (SLM Class) 272
Adding an SLM Class to the Catalog 272
Applying an SLM Class to a Virtual Server 273
Applying SLM Classes from TrafficScript 273
SLM Class TrafficScript Examples 273
"FrontPage Scripts Only" Service Level Monitoring 274
Prioritizing Resources with Service Level Monitoring 274
Graphing SLM Class Conformance Rates 275

CHAPTER 19 Content Caching 276

Introduction 276
Configuring Content Caching 276
Applying Content Caching to a Virtual Server 277

Configuring Lifetimes 277
Configuring Web Cache Memory Usage....278
Monitoring Cache Activity 280
Configuring Disk-Based Caching 280
Caching Policy....281
Requests....281
Responses 281
Variants....282
Byte Ranges....282
ETags....283
Controlling Content Caching Using TrafficScript 283
HTTP Request Processing....283
HTTP Response Processing 283
TrafficScript Cache Control Functions....283
Forcing Stale Content out of the Cache 285
Manual Removal of Cached Content 285
Programmatic Removal of Cached Content....286

CHAPTER 20 Aptimizing Your Web Content 287

Introduction....287
Modes of Operation....287
Configuring Aptimizer for Your Services 288
The Aptimizer Wizard 289
Application Scopes 289
Aptimizer Profiles....290
Measuring Aptimizer Changes....295
Checking That Aptimizer Is Active....295
Using Stealth Mode to Test Aptimizer....295
Measuring Web Page Speed 296
Tools....296
Understanding Custom Acceleration Profiles....298
Acceleration Settings 300
Understanding Optimization Techniques 305
Web Page Speed Rules 306
Resource Naming and URL Versioning 309
Using a Content Distribution Network 313
Troubleshooting Aptimizer....314
Controlling Unexpected Behavior 314
Interaction with Other Traffic Manager Functionality....315
Runtime Errors 318
Image Errors 319
CSS Errors 321

JavaScript Errors....321
Other Configurable Global Settings....323

CHAPTER 21 Event Handling and Alerts....324

Overview....324
Event Types....325
Creating New Event Types....326
Actions....327
Testing Actions....328
Configuring an Event Handler....329
Duplicate Events....329
Custom Actions....329
Calling a Program or Script....329
Sending a SOAP Message....330
Raising Events from TrafficScript or Java Extensions....332
Example....333

CHAPTER 22 Configuring System Level Settings....335

Network Configuration....335
Configuring the Hostname and IP Addresses....335
Configuring VLANs....336
Configuring Your DNS Settings....337
Configuring Routing....338
Configuring Return Path Routing....338
Configuring IP Forwarding and Network Address Translation (NAT)....342
Time and Date Configuration....345
Setting the Time Manually....345
Using an NTP Server....345
Synchronizing Time from the Traffic Manager....346
Remote Login to the Traffic Manager....346
Entering Custom Kernel Parameters....346
Adding or Modifying a Parameter....346
Existing Entries....347
Integration with Riverbed Cloud SteelHead....347

CHAPTER 23 System Security....351

Firewall and Operating System Settings....351
Firewalling Techniques....351
Firewall Configuration with the Traffic Manager....351
Network Design....352
UNIX User Permissions....353
File System Security....354

Operating System Settings 354

CHAPTER 24 Admin Server Security 356

Basic Administration Server Settings 356
Changing the Admin Server SSL Certificate 356
Restricting Access to the Admin Server 357
Changing Admin Server Ports 357
Traffic Manager SSH Server Security 357
Cluster Communication 359
SSL Settings for Admin Server and Internal Connections 360
Access to the REST API 361
User Management 361
User Authentication 361
Local Users 362
Authenticators 364
Testing an Authenticator 368
Permission Groups 369
Login Timeout 370
Suspended Users 370
Login Security and Behavior 370
The Login Information Banner 372
The Event and Audit Logs 372

CHAPTER 25 The Traffic Manager Control API 374

Introducing the Traffic Manager Control API 374
Example: Listing Running Virtual Servers 374
Perl with SOAP::Lite 374
C Sharp or Mono 375
Further Examples 377

CHAPTER 26 Command Line Interface 378

Accessing the CLI 378
Permissions 379
Commands 380
Control API methods 381
Built-in Commands 384
Scripting the CLI 388
Script Output 389

CHAPTER 27 Granular Configuration Import/Export with zconf 390

Introduction 390

Using zconf 390
Exporting a Complete Backup 391
Configuration Listings 392
Partial Imports 392

CHAPTER 28 Multi-Site Cluster Management 393

Introduction 393
Activation and Deactivation 394
Key Concepts 394
Configuration Locations 394
Clusters 395
Deployment Scenarios 395
Create and Manage a Second Traffic Manager Location 395
Add a New Traffic Manager to Your Multi-Site Cluster 396
Merging Two or More Existing Traffic Manager Clusters 396
Configuration 398
Setting Up Locations 398
Setting Traffic Manager Locations 398
Location-Specific Configuration 399
Home Page Changes 400
The World Map 401
Traffic Visualization 401

CHAPTER 29 The Traffic Manager DNS Server 402

DNS Primer 402
Introduction 402
The Layout of DNS 402
Delegation of Authority 402
Name Resolution 403
Resource Records 403
Zone Files 404
The Resolution Process 406
Supported DNS Features 407
Implemented Features from RFC 1034 407
Implemented Features from RFC 1035 408
Exceptions for RFC 1034 409
Exceptions for RFC 1035 409
Other Implemented Features 409
Other Excluded Features 410
Configuring the DNS Server 411
Configuration Summary 411
Uploading DNS Zonefiles to the Traffic Manager 412

Setting Up Traffic Manager Zones 412
Configuring a DNS Virtual Server 412

CHAPTER 30 Global Load Balancing 415

Introduction and Prerequisites 415
About Global Server Load Balancing 416
GSLB Within the Traffic Manager 417
Deployment Planning 417
Traffic Manager Positioning 417
Deployment Methods 418
The Time-to-Live (TTL) Field 423
Components of a Traffic Manager GLB Deployment 423
GLB Locations 423
GLB Services 424
GLB Configured Virtual Servers and Pools 425
DNS Servers 425
Service IP Addresses 425
Service Monitors 425
Configuring GLB 426
Overview 426
Defining GLB Locations 426
Creating a Service Monitor 427
Creating a GLB Service 428
Creating a DNS Server Pool 433
Creating a DNS Virtual Server 433
Traffic Visualization 434
The Current Activity Graph 434
The Historical Activity Graph 435
The Connections Page 435
GLB Request Logs 435
Testing DNS with DIG 435
Extending the Traffic Manager's GeoIP Database 436
Unrecognized IP Addresses 436
Extending the Traffic Manager's GeoIP Database 437
Testing the IP Address Mappings 437
Updating Your Traffic Manager Cluster Configuration 438

CHAPTER 31 FIPS Validation in the Traffic Manager 439

Introduction to FIPS 439
FIPS Mode 439
FIPS 140-2 439
FIPS 140-2 and the Traffic Manager 440

Deploying FIPS Mode 443
Preparation 443
Enabling FIPS Mode 447
Operating in FIPS Mode 448

CHAPTER 32 Kerberos Constrained Delegation Support 450

The Kerberos Protocol 450
Kerberos Protocol Transition and Constrained Delegation 450
Protocol Transition 450
Constrained Delegation 450
Rationale for Using Kerberos 451
Configuring Kerberos on the Traffic Manager 451
Traffic Manager Service Principal 451
Virtual Server Protocol Transition Configuration 453
Pool Protocol Transition Configuration 454

CHAPTER 33 Troubleshooting 456

Tools and Techniques 456
Diagnosis and Event Logging 456
Monitoring Requests and Responses 457
Connection Activity Report 458
Request Logs 458
Advanced Logging 459
Monitoring Events 459
Detailed Debugging of Connections 459
Testing Individual Nodes 461
Understanding Your Configuration 461
Troubleshooting Tips 462
Generating Test Requests 462
Checking Automatic Back-End Failover 463
Checking Automatic Front-End Failover 463
Common Problems 464
Did Not Become Root 464
Connection Refused 464
Inappropriate Traffic IP Addresses Configured 465
The Traffic Manager Drops Connection Before Protocol Begins 465
Web Server Returns Error 400 465
Wrong Port Number Configured 465
Running Out of File Descriptors 466
Running Out of Disk Space 466
Getting Help 467

CHAPTER 34 Glossary 468

CHAPTER 35 Software License Acknowledgements 475

License for the Berkeley DB Code (Version 1.85) 475
RSA PKCS11 476
License for the OpenLDAP Code, Version 2.4.23 476
PCRE License 477
Libnet License 478
License for Yahoo! UI Library 479
License for ssleay Cryptographic Library 479
License for libxml2 and libxslt 481
License for the Java Servlet API 481
License for the Expat XML Parser 482
License for MooTools 483
Licenses for OpenLayers 483
License for rsync 485
License for mod_imap.c 486
License for Antlr and libantlr 487
License for es3-grammar 488
License for jsoncpp 489
License for libjpeg 489
License for libunwind 491
License for the Perl JSON Library 492
License for OpenSSL 494
License for WebP 497
License for Flex 498
License for CryptoJS 499
License for zlib.js 500
License for XML::Twig 501
License for MIT Kerberos 501
License for Libedit 525
License for ZebOS 527
License for Curl 527
License for Jansson 528
License for Digest::SHA 529
License for Sys::SysLog 529

CHAPTER 36 Index 530

Preface

Welcome to the *Brocade Virtual Traffic Manager: User's Guide*. Read this preface for an overview of the information provided in this guide and the documentation conventions used throughout, additional reading and contact information. This chapter includes the following sections:

- "About This Guide", next
- "Documentation and Release Notes" on page 19
- "Getting Technical Help or Reporting Errors" on page 20

About This Guide

The *Brocade Virtual Traffic Manager: User's Guide* describes how to configure and manage the Brocade Virtual Traffic Manager (Traffic Manager).

Brocade recommends first reading the *Brocade Virtual Traffic Manager: Installation and Getting Started Guide* applicable to your product variant for an introduction to installing the Traffic Manager and performing basic configuration to load-balance services.

This document describes the features and capabilities of the Traffic Manager release 10.1.

Audience

This guide is written for system administrators familiar with administering and managing Web services and infrastructure.

This guide assumes you are familiar with networking terminology.

Document Conventions

This manual uses the following standard set of typographical conventions to introduce new terms, illustrate screen displays, and describe command syntax.

Convention	Meaning
italics	Within text, new terms and emphasized words appear in italic typeface.
boldface	Within text, CLI commands and UI controls appear in bold

	typeface.

`Courier`	Code examples appear in Courier font:
	`amnesiac > enable`
	`amnesiac # configure terminal`

<>	Values that you specify appear in angle brackets: **interface <ipaddress>**
[]	Optional keywords or variables appear in brackets: **ntp peer <addr> [version <number>]**
{}	Required keywords or variables appear in braces: **{delete <filename> | upload <filename>}**
|	The pipe symbol represents a choice to select one keyword or variable to the left or right of the symbol (he keyword or variable can be either optional or required): **{delete <filename> | upload <filename>}**

Documentation and Release Notes

To obtain the most current version of all Brocade documentation, click through to the desired product page on the Brocade Web site at http://www.brocade.com/en/products-services.html.

If you need more information, see the Brocade Knowledge Base for any known issues, how-to documents, system requirements, and common error messages. You can browse titles or search for keywords and strings. To access the Brocade Knowledge Base, login to the MyBrocade Web site at https://login.brocade.com.

Each software release includes release notes. The release notes identify new features in the software as well as known and fixed problems. To obtain the most current version of the release notes, login to the MyBrocade Web site at https://login.brocade.com.

Examine the release notes before you begin the installation and configuration process.

Traffic Manager Documentation

The Traffic Manager includes a comprehensive user's guide that describes the Traffic Manager's feature set in depth. There are also getting started guides for each product variant, and a series of reference guides to cover additional functionality such as the TrafficScript rules language and product APIs.

You can download documentation for all supported versions of the Traffic Manager from the Brocade Virtual Traffic Manager product page on the Brocade Web site at:

http://www.brocade.com/en/products-services/application-delivery-controllers/virtual-traffic-manager.html.

Traffic Manager Online Help

Click the **Help** button on any page of the Admin UI to see detailed help information for that page. You can also view contents and use index pages to navigate the online help.

The Rules > Edit page also has a link to TrafficScript Help, a quick reference guide for all Traffic Manager TrafficScript functions.

Traffic Manager Information Online

For Traffic Manager product specifications, see:

http://www.brocade.com/en/products-services/application-delivery-controllers/virtual-traffic-manager.html

Visit the Brocade Community Web site for further documentation, examples, white papers, and other resources:

http://community.brocade.com/

Getting Technical Help or Reporting Errors

Brocade is committed to ensuring that your investment in our products remains cost-effective. If you need assistance or find errors in the documentation, contact Brocade using one of the following options.

Web Access

The Brocade Web site contains the latest version of this guide and all other user guides for the Traffic Manager. For more information, see http://www.brocade.com/en/products-services/application-delivery-controllers/virtual-traffic-manager.html.

To report errors, log in to the MyBrocade Web site at https://login.brocade.com and click **Support Cases** to open a new support case. Make sure you specify the document title in the case description.

E-mail and Telephone Access

Go to http://www.brocade.com/en/support.html for the latest e-mail and telephone contact information.

CHAPTER 1 Traffic Manager Overview

Introducing the Traffic Manager

The Traffic Manager product family provides high-availability, application-centric traffic management and load balancing solutions. They provide control, intelligence, security and resilience for all your application traffic.

The Traffic Manager is intended for organizations hosting valuable business-critical services, such as TCP and UDP-based services like HTTP (Web) and media delivery, and XML-based services such as Web Services.

The Traffic Manager's unique process architecture ensures it can handle large volumes of network traffic efficiently. Its inherent scalability allows you to add more front-end Traffic Managers or back-end servers to your cluster as the need arises. The cluster size is unlimited, and the performance of the Traffic Manager grows in line with the performance of the platform used.

The Traffic Manager represents a family of highly capable solutions that can be adapted and extended as new requirements arise. Using the unique TrafficScript language and built-in Java Extensions you can write sophisticated, tailored traffic management rules to inspect, transform, manage and route requests and responses. TrafficScript rules can manage connections in any TCP or UDP-based protocol.

Traffic Manager products are secure out-of-the-box, and are hardened against intrusion and Denial-of-Service (DoS) attacks. They incorporate the fastest and strongest SSL encryption technologies, and can efficiently decrypt and encrypt large numbers of SSL connections. TrafficScript rules, security policies and other content-based calculations can be applied to the encrypted request while retaining full end-to-end security.

For critical, high-availability solutions, The Traffic Manager offers cluster redundancy. This allows you to have unlimited numbers of active and passive standby front-end servers. If one of your active machines fails, a standby server is automatically brought into action; in the case of subsequent failure, more standby servers are available to take up the load. This ensures that there is no single point of failure in the system.

Typical Deployment

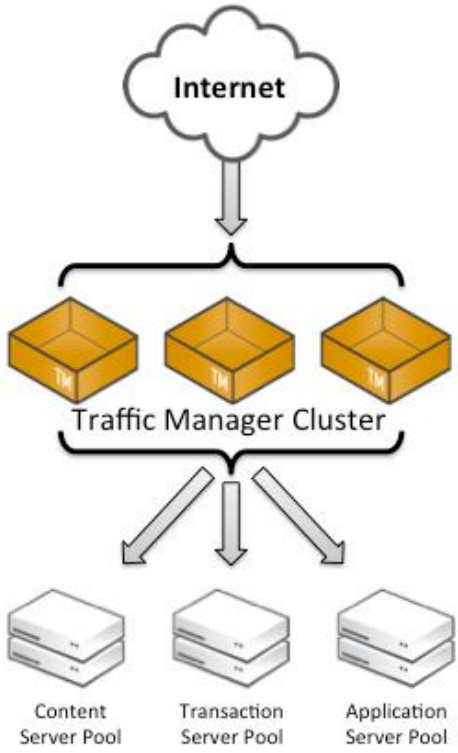


Fig 1. A typical deployment using a cluster of Traffic Managers

Traffic Manager Product Variants

The Traffic Manager product family is available in a variety of software, virtual appliance and cloud instance configurations. All variants share the same core Traffic Manager software, but different versions may provide different levels of functionality depending on the enabling license key.

This manual documents the full functionality of the Traffic Manager software with all options enabled. It may describe features and capabilities that are not present or visible in the version of the product you are using. Features present but not enabled in your license key will be greyed-out and un-selectable in the Admin UI.

For example, Global Load Balancing, Service Level Monitoring, Rate Shaping, Autoscaling and Bandwidth Management are examples of advanced product capabilities and may not be enabled in your particular configuration.

In addition, Brocade provides two optional Traffic Manager components, available only through an appropriate license key upgrade:

- **Brocade Virtual Web Application Firewall**: An implementation of a Web application firewall, used for providing advanced attack detection and protection

for your Web applications. See "The Web Application Firewall" in CHAPTER 7 for more details about how this fits into your Traffic Manager infrastructure. For full product details and instructions, see the *Brocade Virtual Web Application Firewall User Guide*, available from the Brocade Support Web site at:

https://www.brocade.com/service-support/

- **Aptimizer**: Provides content optimization functionality for your Web applications. This is available as either a fully integrated component of the Traffic Manager, or in standalone proxy mode whereby the load balancing aspects of the Traffic Manager are disabled. Your sales representative can provide details about which variant is most appropriate for your needs. CHAPTER 20, "Aptimizing Your Web Content", provides full details about how to enable and configure Aptimizer for your infrastructure.

  Note: Aptimizer functionality is not available for software variants running on SPARC-based Solaris or SunOS platforms.

Appliance and cloud versions of the Traffic Manager feature Networking and Date/Time configuration options not available in software-only versions.

Your product version specifications describe which capabilities are enabled in your particular variant. See also the applicable installation and getting started guide available from the Brocade Web site.

Developer Mode

When unlicensed, The Traffic Manager falls back to a default state known as Developer mode. This is designed to allow the user to experience the full features and capabilities of the Traffic Manager for development or evaluation purposes. Full product functionality is provided, but in a bandwidth-constrained environment. It operates with a maximum bandwidth limited to 1Mb/s and 100 SSL TPS (transactions per second).

Important: The Developer mode is not designed or intended for full production use. It is recommended that you contact your support provider for details about how to purchase a license key suitable for your needs.

Supported Platforms

The Traffic Manager software can be deployed on a range of platforms, on physical or virtual servers, and in cloud infrastructures. Please refer to the release notes and documentation at http://www.brocade.com/en/products-services/application-delivery-controllers/virtual-traffic-manager.html for up-to-date platform and version number requirements.

Software

The Traffic Manager is available as a software-only package suitable for deployment on existing Linux/UNIX servers. Supported distributions are listed in the release notes as mentioned above.

Virtual Appliances

Brocade provides Traffic Manager virtual appliance packages for VMware vSphere, Citrix XenServer, OracleVM, Microsoft Hyper-V, and QEMU/KVM.

Cloud Computing Services

The Traffic Manager is supported with Amazon's Elastic Compute Cloud (EC2), Rackspace and Microsoft Azure. For EC2, The Traffic Manager is available as an Amazon Machine Image (AMI) and standalone software package for EC2 Linux/UNIX virtual machines.

Supported Cluster Combinations

As discussed in previous sections, The Traffic Manager is available as a number of product variants on a selection of platforms. In addition to the core Traffic Manager software, various product options and capabilities are available through a suitable license upgrade.

When clustering multiple Traffic Managers together for high-availability and redundancy, it is possible to take advantage of The Traffic Manager's flexible architecture and host instances on different platforms within the same cluster. However, some care must be taken to ensure that each cluster member is running the same product configuration.

The following should be taken into account when planning your cluster:

- Mixing different supported host operating systems for the software variant is supported and should not adversely affect cluster operations.
- Although mixing license key types IS technically possible within the cluster, it is not recommended. There are likely to be warnings and/or errors within the Admin UI if features only licensed on a sub-set of the cluster are used. This is due to the automatic config replicator attempting to apply non-licensed functionality across the cluster.
- Cloud instances (for example, the Traffic Manager on Amazon EC2) cannot directly be clustered with non-cloud instances (software, virtual appliance,

hardware appliance). However, the *multi-site cluster management*[1] feature enables centralized control of multiple Traffic Manager clusters regardless of location.

- The Web Application Firewall feature cannot be used in a mixed-platform cluster (for example, software and virtual appliance).
- Having different product versions within a cluster is unsupported, except in the case of performing an upgrade (or downgrade) of the entire cluster. During this transient state, some Traffic Managers will report that they are running different versions to others in the cluster. This is to be expected, and you should find that such error conditions are cleared once all cluster members have been upgraded.

Important: If you are in any doubt as to the potential effects of running your proposed Traffic Manager infrastructure, Brocade recommends contacting your support provider for assistance.

Chapter Outline

CHAPTER 2 Network Layouts

This chapter discusses the network configuration needed for a typical traffic-managed server farm. It covers hardware requirements, the layout of the network, IP addresses and DNS entries.

CHAPTER 3 Initial Configuration

This chapter explains the concepts of the Traffic Manager architecture, including its system of *Virtual Servers*, *Pools* and *Rules*. It describes how to start managing your first service.

CHAPTER 4 Virtual Servers

Virtual servers are one of the two fundamental configuration concepts in the Traffic Manager software. This chapter describes virtual servers in more detail.

CHAPTER 5 Pools

A pool is the second of the two fundamental configuration concepts in the Traffic Manager software. This chapter describes pools in more detail.

[1] See CHAPTER 28 for more details.

CHAPTER 6 Traffic IP Groups and Fault Tolerance

Fault Tolerance-resilience to failures of back-end servers or individual Traffic Manager systems-is covered in this chapter.

CHAPTER 7 Key Features in the Traffic Manager Administration Interface

Now that you have a basic Traffic Manager system running, this chapter covers the Traffic Manager's performance monitoring and diagnosis functions and other tools in the Traffic Manager Administration Interface.

CHAPTER 8 TrafficScript Rules

This chapter introduces the TrafficScript language, used for writing rules to manage your traffic. It explains how to create rules and gives some examples. A separate **TrafficScript Guide** describes the Traffic Manager's TrafficScript capability in full detail.

CHAPTER 9 TrafficScript Authentication Support

This chapter provides details about remote authentication support through TrafficScript. This can be used to add remote user verification to your virtual servers.

CHAPTER 10 Java Extensions

This chapter describes the Traffic Manager's Java Extensions, allowing you to invoke Java code from TrafficScript. A separate Java Development Guide describes the Traffic Manager's Java Extensions capability in full detail.

CHAPTER 11 Protocol Support

Some protocols can benefit from specific support, and this chapter describes special protocol handlers in the Traffic Manager.

CHAPTER 12 Session Persistence

This chapter describes the Traffic Manager's session persistence features. It outlines some useful session persistence strategies and explains how it is set up within the product.

CHAPTER 13 SSL Encryption

This chapter deals with SSL decryption of incoming traffic, and encryption of requests being sent to the back-end servers. It describes how to set up SSL certificates, authorities and revocation lists within the product and how to apply them to your services.

CHAPTER 14 Health Monitoring

This chapter discusses the Traffic Manager's Health Monitoring capabilities. Health Monitors are used to test the correct operation of back-end server nodes; in the event of a failure, they cause the Traffic Manager to raise an alert, and to route traffic away from the failed node.

CHAPTER 15 Service Protection

This chapter describes the service protection capabilities of the product. It explains how a class of service protection settings can be defined in the catalog, and used by one or more services.

CHAPTER 16 Bandwidth Management

This chapter describes the bandwidth management capabilities of the product. It explains how a class of bandwidth management settings can be defined in the catalog, and used by one or more services.

CHAPTER 17 Request Rate Shaping

This chapter describes how a Traffic Manager can be used to rate-shape requests to your servers and applications to restrict users' activities and prevent applications from being overwhelmed by requests.

CHAPTER 18 Service Level Monitoring

This chapter describes the service level monitoring of the product. It explains how a class of service level monitoring settings can be defined in the catalog, and used by one or more services.

CHAPTER 19 Content Caching

This chapter describes the Traffic Manager's HTTP Content Caching capabilities. You can cache commonly requested Web resources and respond to subsequent requests directly from the cache, rather than forwarding many identical requests to the back-end server nodes.

CHAPTER 20 Aptimizing Your Web Content

This chapter covers the Aptimizer and how you can apply Web content optimization to your Web services.

CHAPTER 21 Event Handling and Alerts

This chapter describes how to control what the Traffic Manager does when particular events occur, and how to inform or drive external management systems.

CHAPTER 22 Configuring System Level Settings

This chapter describes additional system level configuration specific to virtual appliance, cloud instance, and certain software variants of the Traffic Manager.

CHAPTER 23 System Security

This chapter gives tips for secure operation of your Traffic Managers. It covers firewalling and network design, and Unix software permissions that are relevant to the Traffic Manager software.

CHAPTER 24 Admin Server Security

This chapter covers security measures to control access to the Administration Interface, including user authentication against external databases.

CHAPTER 25 The Traffic Manager Control API

This chapter gives a brief introduction to the Traffic Manager's Control API. For full information, see the separate *Brocade Virtual Traffic Manager: Control API Guide*, available from the Brocade Web site.

CHAPTER 26 Command Line Interface

This chapter covers the Command Line Interface in detail, with reference to the Control API methods defined in the *Brocade Virtual Traffic Manager: Control API Guide*.

CHAPTER 27 Granular Configuration Import/Export with zconf

This chapter describes the `zconf` command line utility that can be used to perform a fine-grained import/export on individual configuration objects.

CHAPTER 28 Multi-Site Cluster Management

Furthering the concept of a cluster of Traffic Managers, this chapter introduces the idea of combining several clusters into one centrally managed *multi-site* super-cluster.

CHAPTER 29 The Traffic Manager DNS Server

The Traffic Manager provides an internal DNS server capability as an alternative to externally sourced DNS. This chapter discusses the implementation and configuration of the Traffic Manager DNS server.

CHAPTER 30 Global Load Balancing

This chapter discusses the implementation of Global Server Load Balancing (GSLB) within the Traffic Manager.

CHAPTER 31 FIPS Validation in the Traffic Manager

This chapter provides details for prospective users looking for information on how FIPS is implemented in the Traffic Manager.

CHAPTER 32 Kerberos Constrained Delegation Support

This chapter discusses the Traffic Manager's implementation of the Kerberos protocol.

CHAPTER 33 Troubleshooting

This chapter describes how to investigate and diagnose problems or unexpected behavior with your load-balanced system.

CHAPTER 34 Glossary

The Glossary defines some terms used in this manual. Some of these terms are used with varying meanings in computing literature, so if in doubt you should refer to this section.

CHAPTER 2 Network Layouts

This chapter discusses the configuration of your network. It describes the hardware you will need for an effective traffic-managed server farm, and the DNS and IP address layout.

Essentials of Network Configuration

The components of a basic traffic-managed server farm are:

- One or more front-end machines running the Traffic Manager software
- A number of back-end servers (such as web or mail servers)

The front-end machines must be able to receive traffic from the Internet (or where the remote clients are located). They must also be able to contact the back-end machines.

The back-end servers will usually be visible only from an internal network. The front-end machines do not need to route traffic between the Internet and the back-end machines.

The Traffic Manager software is commonly deployed on a multi-homed machine. One network interface card is visible to the Internet; one or more network interface cards are exposed to the internal private networks where the back-end servers reside. It is also easy to configure a Traffic Manager on a machine with a single network card (this is common in an evaluation or testing environment), where a Traffic Manager can contact both the clients and the servers.

A fully fault-tolerant set-up will contain two or more front ends and several back-end servers. If any one machine fails, the Traffic Manager's failover capability ensures that requests are routed to other machines, ensuring there is no single point of failure in the system.

Note: Some product versions are restricted to just a cluster size of two Traffic Manager machines. Larger cluster sizes can be used with a software key upgrade.

If hardware availability is limited, fewer servers can be used. In the minimal case, it is possible to install the traffic management software and an Internet service on the same machine. This is not recommended, as it reduces the usefulness of the product and the ability to provide fault tolerance in the event of a hardware failure. It may, however, be useful for evaluation or demonstration purposes.

The Traffic Manager can be used in conjunction with a stand-alone firewall. The Traffic Manager machines should be visible both from the Internet and from the internal network; they should probably be put in the DMZ.

Note: CHAPTER 23 discusses the security aspects of network setup in more detail. You should read this section carefully before setting up live services.

Dedicated Management Network

When you configure a Traffic Manager, you can choose to nominate a single management IP address. The Traffic Manager then only accepts management traffic on that address. See the installation and getting started guide for the configuration procedure required to set a dedicated management IP address on your product variant.

For Traffic Manager documentation, see the Brocade Web site at: http://www.brocade.com.

Management traffic includes all access to the Web-based Traffic Manager Admin UI, the Control API, the REST API, and any configuration or state sharing within a Traffic Manager cluster.

Use can use the management IP address to provide a dedicated, trusted management network. Typically, each host running a Traffic Manager has a dedicated network card that is connected to the management network.

Note: For more information on the security aspects of network setup, see "System Security" on page 351. Brocade recommends you read this chapter carefully before setting up live services.

To modify the management IP address on a fully configured Traffic Manager, use the **System > Traffic Managers** page of the Admin UI. For software variants, you can also rerun the "configure" script. Note that a software restart is required for this procedure.

Important: Each management IP address is a single point of failure in a Traffic Manager cluster. If the management network fails, all inter-machine communication is lost and remote configuration using the Admin UI, Control API or REST API is impossible.

For resilience, the fault-tolerance messages that each Traffic Manager sends are broadcast over all network cards. You can restrict this traffic to the management network in the Admin UI.

Sizing Your Cluster

Back-End Servers

The Traffic Manager can support an unlimited number of back-end servers. These might include Web servers, mail servers and FTP servers. They can run any software on any platform to provide the services the Traffic Manager manages.

The servers can be arranged into *pools* to serve different types of content. For instance, you can have several groups of Web servers, including Windows machines running IIS™ and UNIX machines running Zeus Web Server or Apache™, to serve different types of static and dynamic Web content. The Traffic Manager's system of pools and rules allows you to classify requests and send them to different groups of servers. Pools and rules are explained in "Initial Configuration" on page 52.

The number of servers you dedicate to a certain function may vary. To serve static Web pages, if requests are light, two Linux machines could be sufficient; while more complex Sun℠ JSP™ or Microsoft ASP pages might be served by larger numbers of machines.

The Traffic Manager allows you to use any number of machines; should your requirements change, you can add or remove back ends dynamically without disrupting your service.

Content Management on Back Ends

If a number of back ends serve the same site, changes in the site content must be propagated to each back-end server. This can be done in one of two ways:

- Each back end can store a local copy of the content, and the data can be synchronized regularly;
- A shared file system can be used, such as an NFS file server. This stores the content, and each back-end server retrieves data when required.

To serve static Web content, either content management method works well. For SMTP or POP3 servers, a shared file system will be needed because files are modified by that service.

Front-End Servers

A simple deployment of Traffic Managers would involve one front-end machine. This gives access to the full functionality of the Traffic Manager software, except for front-end fault tolerance. In other words, the Traffic Manager machine is a potential single point of failure.

For a fully fault-tolerant set-up, two or more Traffic Managers should be used. Their IP addresses can be arranged into a *Traffic IP Group* which provides fault tolerance if one machine should fail. Traffic IP groups are described in CHAPTER 6.

A common network deployment is to have the front-end machines on two separate networks: one to communicate with the Internet, and one with the back ends (probably a private network). It is possible, however, to deploy front ends and back ends on just one network. You should ensure that all machines routable from the Internet are appropriately firewalled. See CHAPTER 23 for a discussion of firewalling and network security.

You may wish to increase throughput by adding extra network cards to your Traffic Managers. Having two network cards can also simplify deploying a Traffic Manager on two separate networks, although it is not necessary for this.

The Traffic Manager is highly scalable. Adding more servers or upgrading your existing hardware will increase the performance of your traffic management correspondingly. If you wish to add more Traffic Manager machines later, this can be done easily without disturbing your existing set-up or interrupting your service.

Note: If you are using a software version of the Traffic Manager product family, you should ensure that the host machine is not overloaded with other applications and services. For example, running a web browser or other GUI tools on the same server will diminish the capacity of the software to manage traffic on the server.

The diagram below shows a typical traffic-managed server farm. The front-end machines have separate front-end and back-end network cards as described above.

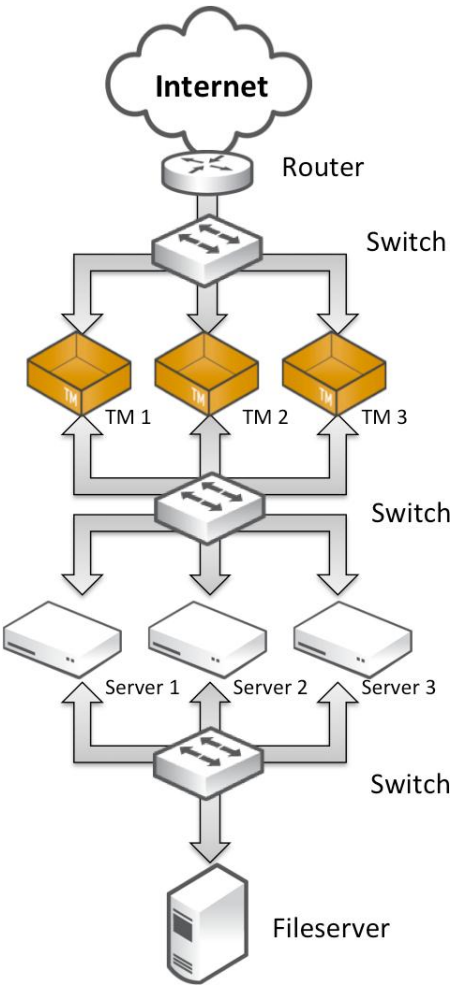


Fig 2. Fault–tolerant network with a cluster of Traffic Manager units serving back-end nodes

Content served by the back ends is held on an NFS file server. Each back-end server has two network cards, one to talk to the Traffic Manager machines and one to talk to this file server, to maximize throughput.

IP Transparency

Your Traffic Manager functions as a full application proxy, terminating network connections from remote clients and making new connections to the selected back-end servers. It does not use the packet-orientated NAT-based load balancing methods that simple layer-4 load balancers use.

With this architecture, the back-end server views the client's request as originating from the Traffic Manager machine, not from the remote client. This can be a disadvantage if the back-end server performs access control based on the client's IP address, or if the server wishes to log the remote IP address. This can often be worked around by performing the access control or logging functions on the Traffic Manager itself, or by making use of the `X-Cluster-Client-Ip` header that the

Traffic Manager inserts into every HTTP connection to identify the client's IP address.

In situations where these workarounds are not appropriate, the Traffic Manager can spoof the source IP address of the server-side connection so that it appears to originate from the client's remote IP address. This capability is known as IP transparency.

IP Transparency can be used selectively. For example, if the Traffic Manager was balancing traffic for a Web farm and a mail farm, you might want SMTP traffic to be IP transparent, but not require that the Web traffic is transparent.

Transparency is enabled on a per-pool basis; you can configure a Web pool that is not transparent, and an SMTP pool that is transparent. The Web pool and the SMTP pool can balance traffic onto the same back-end nodes, or different nodes.

IP Transparency functionality is available on all versions of the Traffic Manager virtual appliance or cloud instance. It is available by default for Traffic Manager software variants installed as root on Linux or UNIX hosts with a kernel version later than 3.2. For kernel versions between 2.6.18 and 3.2, you can gain support for transparency by downloading and installing an additional software module. For further information, see the IP Transparency documentation at the Brocade Community Web site (http://community.brocade.com).

Note: The downloadable IP Transparency module not supported on Linux kernels earlier than version 2.6.18. It is also not available for Solaris/SunOS software variants.

Routing Configuration

Each server node that receives transparent traffic must route its responses back through the Traffic Manager that sent it. This is achieved by configuring the default route on the node to be one of the back-end IP addresses of the Traffic Manager. Please refer to your operating system documentation for details about configuring the default route.

When the server node replies to a request, it will address its response to the remote client IP. Provided the routing is configured correctly, the originating Traffic Manager will intercept this traffic, terminate the connection and process it as normal. The Traffic Manager will then forward the (possibly modified) response back to the client.

It is normally appropriate to configure the Traffic Manager to simply forward on all other packets that are not addressed to it. This will allow the system to function as a local router, so that servers using it as their default route can then contact other systems on nearby or remote networks. The Traffic Manager will need to NAT any packets that it forwards from back-end nodes on private networks.

Note: Refer to your operating system documentation to configure "IP Forwarding" on a host system running the Traffic Manager software variant. See CHAPTER 22 to configure this behavior on a Traffic Manager virtual appliance or cloud instance.

Local Routing Problems

If you use IP transparency, clients on the same network as the back-end nodes will not be able to balance traffic through the Traffic Manager to those nodes.

This is because the back-end server nodes always attempt to reply to the source IP address of the connection. If the source IP address (the client's IP) resides on a local network, the server nodes will attempt to contact the client directly rather than routing via the Traffic Manager system that originated the connection. The connection will appear to hang.

In this case, it might be appropriate to segment the back-end network so that, from the server nodes' perspective, the clients appear to reside on a separate subnet, which must be reached via the default gateway (the Traffic Manager system). Alternatively, a TrafficScript rule could selectively set the source IP address of the server connection to an IP on the Traffic Manager system if the client lies on the same network as the server nodes.

IP Transparency and Traffic Manager Clusters

IP routing is more complex in the case where a cluster of Traffic Managers is used, because each server node can only route back through one IP address.

See "Traffic IP Addresses and Traffic IP Groups", below, for recommendations in this situation.

Traffic IP Addresses and Traffic IP Groups

In a typical network, your back-end servers will be arranged in a local network, with local IP addresses, and will not directly contactable from the Internet. The front-end Traffic Manager machines will have externally available IP addresses, and will be able to connect to the back-end machines over the local network.

The front-end machines will have permanent IP addresses on each network, with the front-end addresses visible and routable from the Internet. These IP addresses are configured by the OS, and if the machine fails, the IP address is lost.

For this reason, these permanent IP addresses are not suitable to use when you publish your services. In the event of a hardware or system failure, your services would become partially or wholly unavailable.

The Traffic Manager's fault tolerance capability allows you to configure Traffic IP addresses. These IP addresses are not tied to individual machines, and the Traffic Manager cluster ensures that each IP address is fully available, even if some of the clustered Traffic Manager machines have failed.

In a typical fault-tolerant configuration, the DNS name used to publish your services is configured to resolve to the traffic IP addresses in your Traffic Manager cluster. This ensures that your services are always fully available.

The traffic IP addresses are arranged into a Traffic IP group. This group spans some or all of your Traffic Manager machines. The machines negotiate between them to share out the traffic IP addresses, each Traffic Manager then raises the IP address (or IP addresses) allocated to it.

Setting up traffic IP groups is described in CHAPTER 6.

If any Traffic Manager machine should fail, the other Traffic Managers in the group detect this. One of them then takes over the failed machine's traffic IP addresses to ensure that the service is uninterrupted.

Note: By default, fault tolerance uses unicast traffic to distribute health information and to manage traffic to multi-hosted Traffic IP Addresses. If you change your configuration to use multicast traffic, the switches that your Traffic Managers are connected to must support IGMP snooping, and messages broadcast to the multicast address used by your Traffic Managers should be forwarded to all Traffic Managers in the cluster.

When you join a Traffic Manager to an existing cluster, a number of tests are conducted automatically to ascertain whether broadcast messages are correctly forwarded.

Traffic IP Address Modes

The Traffic Manager supports several modes:

- **Single-hosted**: Traffic IP addresses are raised on a single Traffic Manager in your fault tolerant cluster. If that Traffic Manager fails, another Traffic Manager will raise that IP address and start accepting traffic.
- **Multi-hosted**: Traffic IP addresses are raised on all of the Traffic Managers in your cluster, using a multicast MAC address that ensures that all incoming traffic is sent to all machines. A custom Linux kernel module is used to evenly distribute the traffic between the working Traffic Managers.
- **Route Health Injection**: Traffic IP addresses are raised "privately" (on loopback) by all participating Traffic Managers, and dynamically advertised into the

adjacent routing domain, using either Open Shortest Path First, version 2, (OSPFv2) or Border Gateway Protocol (BGP) as the routing protocol. In response, routers direct traffic to the active Traffic Manager. See "Route Health Injection" on page 45.

Enabling Multi-Hosted Traffic IP addresses imposes a performance hit due to the additional packet processing required by the Traffic Managers in your cluster. Empirical tests indicate that CPU utilization will increase by 25-30% at moderate traffic levels (10,000 requests per second), with a corresponding limit on top-end capacity.

Note: Multi-hosted IP functionality is available on all versions of the Traffic Manager virtual appliance. It is not included by default with the Traffic Manager software variant; however, you can download and install it as an additional kernel module. It is supported on Linux kernels, version 2.6.18 and later. Multi-hosted IP functionality is not currently available for Solaris/SunOS software variants.

For further information regarding supported versions, see the Brocade Community Web site at http://community.brocade.com.

Example Configurations

These configurations assume that you have two Traffic Managers in your cluster, but can be extended if your cluster contains three or more Traffic Managers.

Active-Passive Configuration – Single-Hosted and Route Health Injection Modes

Suppose your Web site's external DNS name maps to the IP address 162.34.64.29. You have two Traffic Manager machines handling traffic for a number of back-end Web servers:

- With single-hosted mode, you can set up a single-hosted traffic IP group spanning both Traffic Manager machines, containing this single IP address. The Traffic Managers will negotiate and one of them will raise the IP address. It handles all the incoming requests. The second Traffic Manager machine is available on standby. If the first machine should fail, the second machine takes over the IP address and starts to manage the traffic.
- With Route Health Injection (RHI) mode, you can set up an RHI traffic IP group spanning both Traffic Managers, containing the single IP address. Set one Traffic Manager as active and the other as passive. Both Traffic Managers advertise the IP address, using an OSPFv2 and/or BGP (depending on your choice of routing protocol) *metric* to express preference (the active Traffic Manager uses a lower metric). The upstream router (typically your default gateway) chooses the lowest metric route, sending all traffic to the active Traffic Manager. If the Traffic

Manager detects a failure, it cancels the advertisement. If the router detects a failure, it disregards that route. In either case, the router switches to sending traffic according to the next-best metric advertisement, which in this case is the passive Traffic Manager.

The advantage of this configuration is that you can be confident that there is sufficient resource in reserve to handle the traffic should one of the two Traffic Managers fail. Debugging and fault resolution is easier when only one Traffic Manager is handling traffic.

Active-Active Configuration – Single and Multi-Hosted Modes

In an active-active configuration, both Traffic Managers manage your traffic. The distribution mode (single-hosted IP or multi-hosted IP) controls how the traffic is shared between them.

With single-hosted mode, you can configure two traffic IP addresses in a traffic IP group, and configure your DNS name to map to the two addresses, such as 162.34.64.29 and 162.34.64.30. The Traffic Managers will negotiate to raise one traffic IP address each. A DNS server can allocate requests to each IP address in turn (round-robin DNS), and each Traffic Manager handles the requests it receives.

If one of the machines fails, the other machine will detect this. It will then raise the failed machine's traffic IP address in addition to its own, and handle all the traffic sent to either address.

With multi-hosted mode, you can continue to operate with one traffic IP address, simplifying your DNS and reducing the number of externally facing IP addresses you require. The traffic IP address is raised on all of the Traffic Managers in the traffic IP group, and incoming traffic to that IP address is shared evenly between the Traffic Managers.

Active-Active with Loopback (Single External Address, Single Hosted Mode)

If multi-hosted mode is not available or prohibited in your infrastructure, you can distribute traffic load to all of the Traffic Managers in your cluster, while still using a single external traffic IP address.

This configuration involves an additional "loopback" virtual server that listens for traffic on the external traffic IP address and then load-balances the requests across the Traffic Managers using their permanent IP addresses.

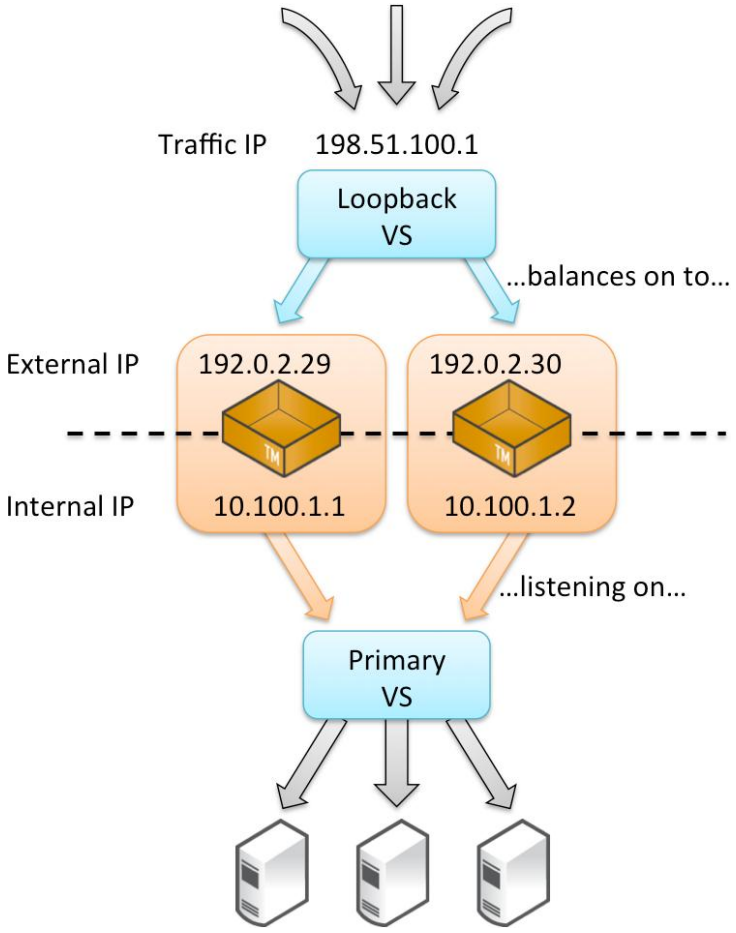


Fig 3. A "loopback" virtual server takes a single traffic IP hosted on one Traffic Manager and distributes it to the rest

First, create your primary virtual server that processes traffic and load-balances it across the back-end servers. Configure this virtual server so that it listens on internal IP addresses and ports on each Traffic Manager, rather than externally accessible ones. For example, the virtual server could listen on 192.0.2.1:80 and 192.0.2.2:80, where the IP addresses are internally visible only.

Then, create a second "loopback" virtual server that listens on the single external IP address and immediately distributes traffic to the primary virtual server across the various Traffic Manager machines in your cluster:

- As in the active-passive example, you should set up a single traffic IP address that will be raised by one Traffic Manager only. Any traffic coming in to this address should then be processed by the simple loopback virtual server, which is listening on that traffic IP address. The loopback virtual server should immediately select a loopback pool that contains the internal IP addresses of the Traffic Manager machines in the cluster; the loopback pool should use a either round-robin or least connections load balancing to evenly distribute traffic across the Traffic Manager machines in the cluster. It should not use a load-balancing method that is influenced by response time, as that will give very uneven request distribution.

The loopback virtual server uses little processing power. Ensure that all of the CPU-intensive processing is performed by the primary virtual server – tasks such as SSL decryption, rules, content compression, and so on.

This method splits the load of more intensive tasks between the two Traffic Managers. If either Traffic Manager fails, the service will continue to run (perhaps with a momentary interruption to some traffic). For example, if the Traffic Manager that is hosting the external traffic IP address were to fail, the traffic IP would be transferred to the remaining Traffic Manager. The loopback pool will detect that one of the nodes was unavailable and direct all traffic to the primary virtual server running on the remaining Traffic Manager.

The target virtual server will observe the connection originating from the loopback virtual server, not the remote client. Generally, the X-Cluster-Client-Ip header can be used to determine the correct source for the connection, but in the common case where SSL requests are forwarded by the loopback virtual server, you should use the **ssl_enhance** and **ssl_trust_magic** settings described in the *Preserving IP Addresses with SSL Forwarding* section in CHAPTER 13.

Multiple-Redundant (N+M) Configuration

In the earlier cases, two Traffic Manager machines are used; if one should fail, a single point of failure is introduced into the system. When running mission-critical services, a higher level of protection can be employed by incorporating several additional Traffic Manager machines to form a multiple-redundant cluster.

Suppose that in normal operation you wish to use *N* active Traffic Managers. You would then incorporate *M* passive Traffic Managers, where *M* is the number of machines that could potentially fail without reducing the number of Traffic Managers in operation.

To achieve this arrangement, you would need *N+M* front-end machines running the Traffic Manager. You can create a traffic IP group containing *N* traffic IP addresses, yet spanning all *N+M* machines. If a machine in your active group fails, a backup machine from the passive group is brought into active duty to take up the load. If another machine fails, an additional passive Traffic Manager becomes active, and so on. This is a typical clustering arrangement incorporating multiple layers of redundancy.

Using IP Transparency with a Cluster

Using IP transparency with a cluster of Traffic Manager machines introduces additional complexity because each server node is configured to route to a single Traffic Manager IP address. However, any of the Traffic Manager machines in the cluster may send transparent connections to the server nodes, and the nodes must route each response back via the Traffic Manager that originated the connection.

Active-Passive Configuration

With a single active Traffic Manager configuration, this can be achieved using the **keeptogether** setting in a traffic IP group that uses single-hosted IP addresses.

Create a traffic IP group containing two IP addresses; the front-end IP address for incoming traffic, and a back-end IP address that resides on the server side network. Select the **keeptogether** option in the traffic IP group.

Configure each back-end server to route all traffic via the back-end IP address you configured in the traffic IP group.

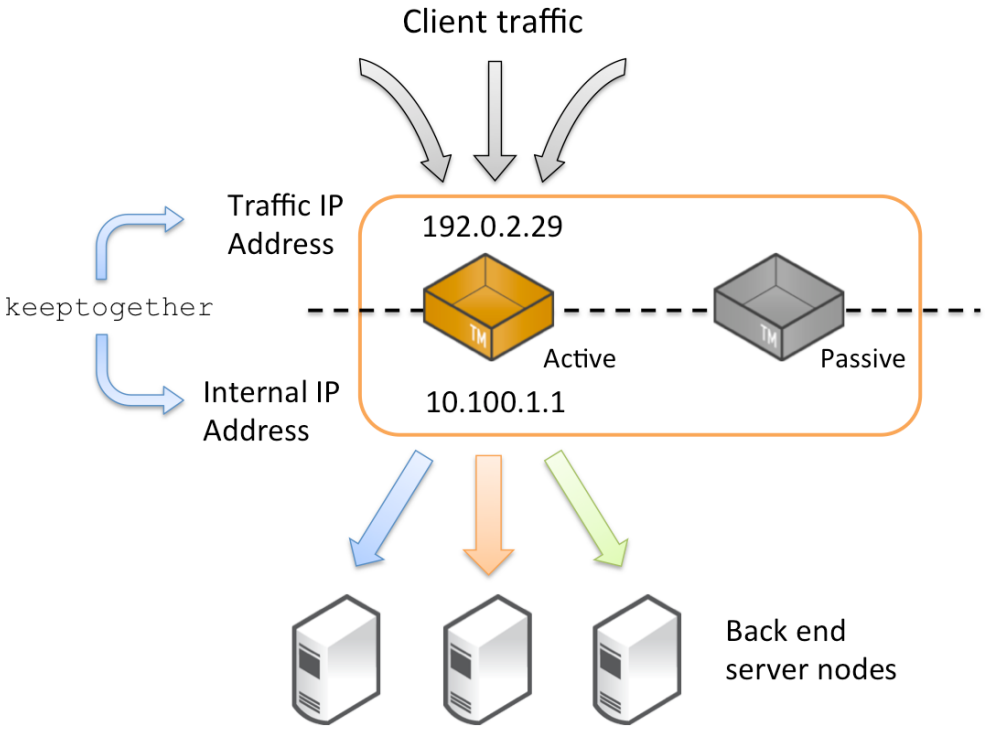


Fig 4. The Traffic IP and Internal IP are bound together in a traffic IP group

With this configuration, both IP addresses will be hosted on the same Traffic Manager machine. If that machine were to fail, both IP addresses would be migrated to the same passive machine, making it the new active machine. The back-end servers will now route traffic back via the new active machine.

Active-Active Configuration

With a configuration involving multiple *active* Traffic Managers, it is necessary to partition your back-end servers into groups, one for each active Traffic Manager machine. These groups should then be defined as pools within the Traffic Manager Admin UI, adding each back-end server as a node accordingly.

All servers in the same pool should have their default route configured to be the back-end IP address of the corresponding Traffic Manager. Please refer to your

operating system documentation for more details about how to manipulate your server route settings.

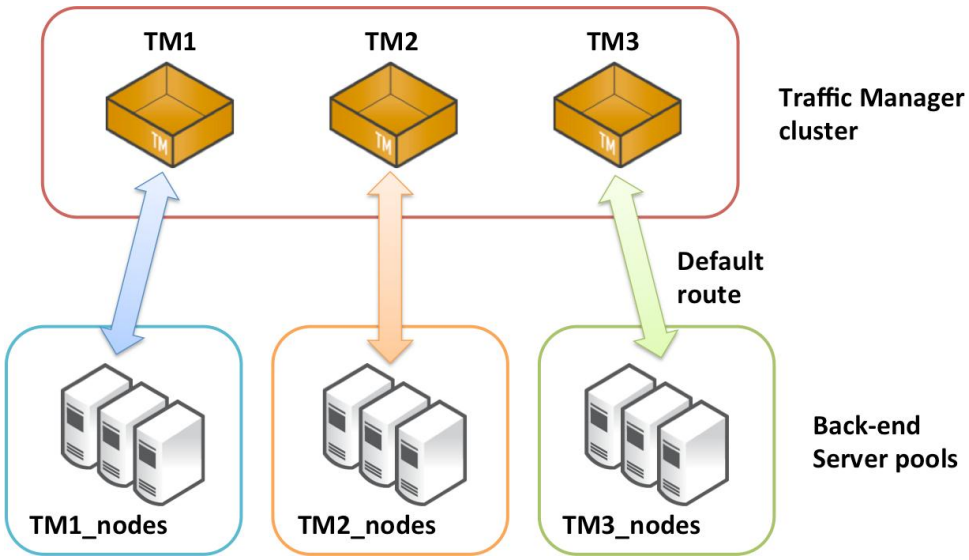


Fig 5. Partitioning your servers into Traffic Manager specific pools

TrafficScript rules can then be used to select the correct pool to route traffic to, based on either of the following:

- The name of the Traffic Manager that is managing that connection
- The local client-side IP address (if using several "keeptogether" Traffic IP Groups in single-hosted mode

The following code snippet demonstrates how this might work (using the Traffic Manager name as the selection criteria):

```
$hostname = sys.hostname();

if( $hostname == "TM1" ) {
   pool.use( "TM1_nodes" );
}
if( $hostname == "TM2" ) {
   pool.use( "TM2_nodes" );
}
if( $hostname == "TM3" ) {
   pool.use( "TM3_nodes" );
}
```

Important: This configuration does, however, include the limitation whereby if a Traffic Manager fails the associated pool will become redundant. Additionally, session persistence cannot reliably be used (particularly if multi-hosted IP addresses are in use).

IP transparency can be used selectively. For example, suppose that a Traffic Manager cluster is managing high volume Web traffic to www.mysite.com, and low volume SMTP traffic to mail.mysite.com. Only the SMTP traffic needs to be transparent. In this case:

- www.mysite.com can resolve to several IP addresses in an Active-Active TrafficCluster configuration without IP transparency.
- mail.mysite.com can resolve to a single IP address using the Active-Passive **keeptogether** configuration described above.

Route Health Injection and the Network

When using Route Health Injection (RHI), the Traffic Manager communicates with routers in the adjacent routing domain. Once it has established communication, the Traffic Manager joins the routing domain and advertises RHI traffic IP addresses into it.

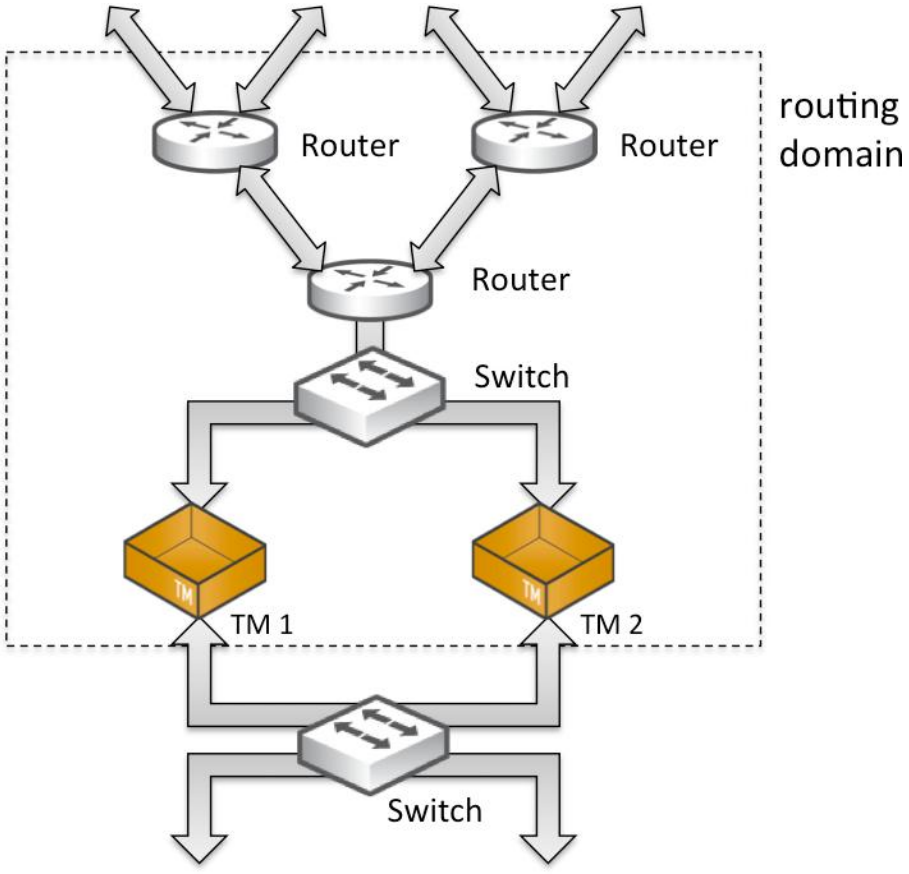


Fig 6. Advertising traffic IP addresses using RHI

Such advertisements are dynamic and respond automatically to your Traffic Manager configuration changes that create, destroy, or move RHI traffic IP addresses. The advertisements also respond automatically to failures detected by the Traffic Manager, and through the routing domain's dynamic routing protocols, to network failures that are not local to the Traffic Manager.

RHI is therefore able to work in a wide variety of deployments. For example, on a small scale to manage simple local failover (such as within a single datacenter rack), and on a large scale to manage traffic distribution and failover between different datacenters within an enterprise or across the whole Internet.

RHI operates using RHI-designated IPv4 traffic IP groups. In a single location, you can use an RHI traffic IP group serviced by either a single *active* Traffic Manager, or an *active-passive* pair of Traffic Managers (See "Active-Passive Configuration – Single-Hosted and Route Health Injection Modes" on page 39).

Note: RHI does not support Traffic IP groups based on IPv6 addresses.

To increase scale, you can repeat this pattern in further Traffic Manager datacenter locations as necessary. Different locations use different RHI traffic IP groups, where the Traffic IP addresses in each group are identical, but the OSPFv2 or BGP metrics used are typically different.

Note: For OSPFv2, this scenario requires all datacenters to be in the same routing domain. For BGP, datacenters can be internal or external to the routing domain.

Locations might have different priorities. For example, with a two-location primary-standby datacenter deployment, you configure the following:

1. In the primary datacenter, configure a primary RHI traffic IP group serviced by an active-passive Traffic Manager pair.
2. In the standby datacenter, configure a standby RHI traffic IP group serviced by an active-passive Traffic Manager pair.
3. Configure the standby RHI traffic IP group with a very high metric, such that the primary datacenter is always preferred unless it is unavailable.

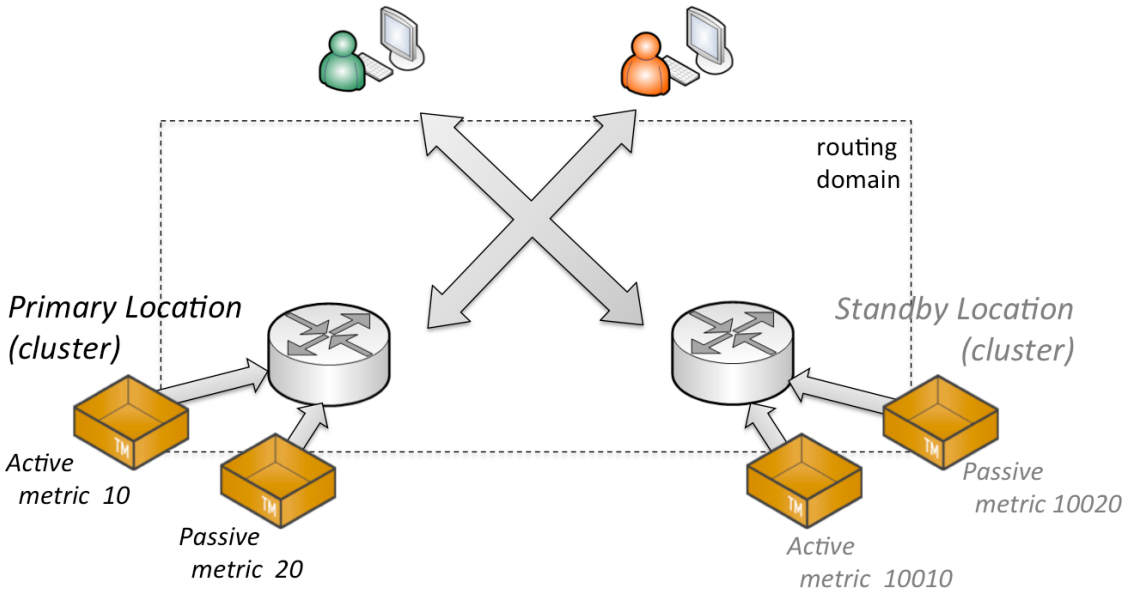


Fig 7. A dual datacenter deployment with RHI

Alternatively, your datacenter locations might have similar priority, resulting in multiple active locations where routing decisions are based on using the best route, according to the network topology. This is often referred to as an *anycast* configuration. To achieve it, configure the RHI traffic IP groups in each location with the same metrics.

Note: RHI implemented with BGP over multiple locations requires other parts of your infrastructure to be configured to support it. In other words, you must ensure that the routers between your datacenter locations respect the supplied metrics, and do not have other policies configured that might influence location choice.

The Credentials Used for RHI Communications

To implement RHI, each Traffic Manager communicates with adjacent routers using OSPFv2 or by establishing a session with BGP. You must configure your Traffic Manager with suitable credentials to enable the Traffic Manager to establish communications with an adjacent OSPFv2 or BGP enabled router. This process is called *peering* with the router.

Note: OSPFv2 communication requires multicast.

You can create clusters of Traffic Managers that use the same credentials. Each cluster (or each Configuration Location within a *Multi-Site Manager* cluster; see "Multi-Site Cluster Management" on page 393) uses one set of credentials, and therefore all Traffic Managers in the cluster (or Configuration Location) join the same area (for OSPFv2) or Autonomous System (AS) (for BGP).

If you have multiple locations that require different credentials, the Traffic Managers in the different locations do not need to be clustered. If, however, you want to cluster them, you must use the Traffic Manager's Multi-Site Manager functionality.

To enter your OSPFv2 or BGP credentials, use the **System > Fault Tolerance** page.

For further details about OSPFv2/BGP configuration and RHI traffic IP group configuration, see "Traffic IP Groups and Fault Tolerance" on page 105.

Troubleshooting RHI Issues

The Traffic Manager uses third party routing software for RHI operations. This routing software logs RHI events to a specific file in your file system: `$ZEUSHOME/log/routing_sw`

The contents of this file are included in your Technical Support Reports (see "Getting Help" on page 467) and can be useful to your support provider in troubleshooting RHI communication problems in your Traffic Manager deployment.

This log file is subject to the following rotation policy:

- For Virtual Appliance and Cloud instances, the Traffic Manager performs log rotation automatically using a fixed file size (50 MB by default). Archived logs are compressed with the "gzip" compression method and stored under a name containing the date and a sequential number.
- For software instances, you must enact your own rotation policy. To inform the Traffic Manager that the log file has been rotated (moved), use the script `$ZEUSHOME/zxtm/zebos/bin/reload_logfile` post-rotation to restart the logging process in a new file.

An Introduction to OSPFv2 and BGP

The Traffic Manager supports Open Shortest Path First version 2 (OSPFv2) and Border Gateway Protocol (BGP) as routing protocols for RHI.

OSPFv2 is an *interior gateway protocol*, typically used to distribute routes inside a single Autonomous System (AS) on a network, for example, with ISPs or large company networks. OSPFv2 enables routers to auto-discover and synchronize with other OSPFv2-configured routers in the same AS.

OSPFv2 works at Layer 3, using raw IP packets rather than over TCP or UDP, using a Time-To-Live (TTL) value of 1. It uses multicast addressing to flood routing information to the next router in the chain, and it is able to handle its own error detection and correction. OSPFv2 is typically internal to a body such as an ISP and is quick to *converge* (when a route changes, convergence is achieved when all routers in a network agree the new quickest route).

For further information on OSPF, see http://en.wikipedia.org/wiki/Open_Shortest_Path_First.

BGP is, by comparison, an *exterior gateway protocol*, typically used to distribute routing and reachability information outside of individual AS's.

Note: BGP can still be used as an interior gateway protocol. For this purpose, the Traffic Manager uses Internal BGP (known as iBGP) for information exchange. For communication between routers in different ASs, the Traffic Manager uses External BGP (known as eBGP).

Unlike OSPFv2, rather than being able to auto-discover their peers, BGP enabled routers require you to define the explicit configuration of the neighbors with which they expect to establish sessions.

For further information on BGP, see www.bgp4.as.

You can configure the Traffic Manager to use either, or both, of these protocols to advertise IP addresses and to have these advertisements propagate through the network. To enable and use Route Health Injection with your services, see "Creating a Traffic IP Group" on page 107.

Introduction to IPv6

IPv6 is a network layer protocol used in switching-packet networks. The main characteristic of IPv6 is the large amount of available addresses, as it uses 128 bits-long addresses instead of the 32 bits length provided by IPv4. It also simplifies the network management avoiding the use of complex subnetting schemes.

Some of the advantages provided by IPv6 are:

- IP security
- Mobile IP addresses
- Simplified header structure
- Address auto-configuration
- Anycast (one address out of many) and mandatory multicast addresses

Main Features of IPv6 in the Traffic Manager

- Acts as a gateway for IPv6
- Can process different IP address versions at your front end and back-end servers

- Able to work in IPv4-IPv6 mixed-networks, and in just-IPv4 networks

IPv6 unicast addresses can be used for configuring your Traffic Manager wherever IPv4 addresses can be used: in traffic IP addresses (excluding RHI and multi-hosted traffic IP addresses), when specifying nodes or the addresses a virtual server is listening on, in TrafficScript rules, and so on.

Your Traffic Manager can also function as a gateway from IPv4 to IPv6 or vice versa and even both at the same time.

Technical Restrictions

Important: Some restrictions apply when using IPv6 in the Traffic Manager environment. Although they should not affect the normal running of the software, these restrictions must always be taken into account.

This is a list of the main restrictions regarding the use of IPv6:

- The internal communication between different Traffic Managers is done over the IPv4 protocol.
- Heartbeat messages only work on IPv4, as does the administration server.
- When using a hostname in the configuration of a back-end node, the Traffic Manager will first look up the IPv4 address. If you want to use the IPv6 address of a machine where the DNS has both IPv4 and IPv6 addresses, you must enter the IPv6 address directly.
- If a host has only an IPv6 address in the DNS, that address will be used.

Tuning Duplicate Address Detection

Note: This section applies to the Traffic Manager software variant only. Traffic Manager virtual appliance/cloud variants are automatically configured with Duplicate Address Detection correctly tuned.

The Duplicate Address Detection (DAD) feature of many operating systems seeks to ensure that two machines do not raise the same address simultaneously. This feature can conflict with the Traffic Manager's fault tolerance; when an IP is transferred from one Traffic Manager system to another, timing conditions may trigger DAD on the Traffic Manager that is raising the address. The DAD feature can be tuned as follows:

- **For FreeBSD:** set the SYSCTL parameter "`net.inet6.ip6.dad_count`" to zero. This can be achieved by editing the system configuration file /etc/sysctl.conf. Add or change the line:

  ```
  net.inet6.ip6.dad_count=0
  ```

This change will take effect after a reboot. To adapt the setting without a system restart, issue the following command:

```
# sysctl -w net.inet6.ip6.dad_count=0
```

Note, however, that this setting will be lost when the system is rebooted.

- **Linux**: the sysctls are called net.ipv6.conf.default.dad_transmits and net.ipv6.conf.all.dad_transmits. Add (or change) these lines in /etc/sysctl.conf to:

  ```
  net.ipv6.conf.default.dad_transmits = 0
  ```

  and:

  ```
  net.ipv6.conf.all.dad_transmits = 0
  ```

  For immediate change, issue this command for each relevant interface (eth1 in this example), plus 'default' and 'all':

  ```
  # sysctl -w net.ipv6.conf.eth1.dad_transmits=0
  # sysctl -w net.ipv6.conf.default.dad_transmits=0
  # sysctl -w net.ipv6.conf.all.dad_transmits=0
  ```

- **Solaris 10:** if your system supports them, change the value of the following settings:

  ```
  # ndd -set /dev/arp arp_probe_count 0
  # ndd -set /dev/ip ip_dup_recovery 50
  ```

CHAPTER 3 Initial Configuration

This chapter explains how to set up a basic traffic-managed site.

Architecture Concepts

A Traffic Manager manages traffic for network services like Web and application servers (HTTP, HTTPS), Web services (SOAP), mail (SMTP, POP, IMAP) and other protocols such as DNS and streaming media.

A ***service*** is published as an IP address and port, and accepts traffic using the appropriate protocol.

A website is hosted at `www.mysite.com`. This DNS name resolves to the IP address 123.123.12.1; the server listens for HTTP traffic on port 80 at this address.

An Internet Service Provider (ISP) publishes its POP3 and SMTP mail servers as `pop.mymail.com` and `smtp.mymail.com`. Each of these servers has a DNS entry linking it to an IP address:

`pop.mymail.com` has address `202.3.45.67`

`smtp.mymail.com` has address `202.3.45.68`

`pop.mymail.com` listens on port 110 for POP3 traffic, while `smtp.mymail.com` listens on port 25 for SMTP traffic.

Within a Traffic Manager, all the traffic for a particular service is handled by a ***virtual server***. This is the interface between a Traffic Manager and the Internet, and is set up for a specified port and protocol; typically, it will manage all the traffic for that protocol.

When the virtual server receives a request, it assigns it to a ***pool***. This is a collection of ***nodes***, each corresponding to a back-end server and port, such as `server1.mysite.com:80`. The pool load-balances traffic across the nodes. You can set up several pools, which may have nodes in common.

To decide which pool to use for a request, the virtual server can apply a list of ***rules***. A rule inspects the incoming request, and decides what action to take with it. It can choose a pool to handle the request, close the request, or pass the request on to the next rule in the list. If no rule makes a positive routing decision, the request is assigned to the virtual server's ***default pool***.

If a node in the pool should fail, the Traffic Manager's ***monitors*** detect this automatically and it stops sending requests to that node. Traffic is distributed among the other nodes in the pool with no visible disruption to the service.

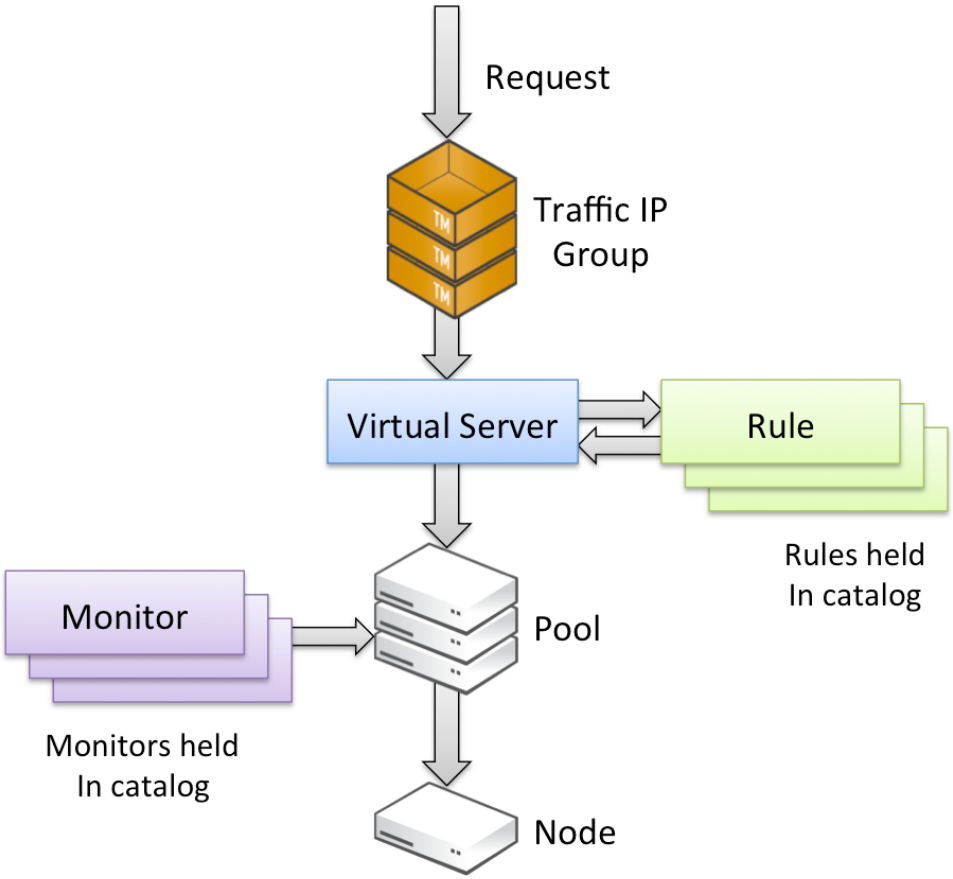


Fig 8. A basic Traffic Manager configuration

A Traffic Manager machine can run many virtual servers, one for each service it manages. Each virtual server can use several rules and pools, and various monitors can watch the pools.

Rules and monitors are organized into ***catalogs***. The Rules Catalog, for instance, holds all the rules which have been written for the system. A rule in the catalog can be applied to any virtual server easily. Other shared items, such as SSL certificates and service protection classes, are also held in catalogs.

Two or more Traffic Manager machines can be arranged in a ***cluster***. Configuration is shared across all the Traffic Manager machines in the cluster; each machine runs the same virtual servers, using the same rules, and so on.

The public traffic IP addresses used for your services can be arranged into ***traffic IP groups***. A traffic IP group spans some or all of the Traffic Managers in the cluster, and these Traffic Managers host the group's IP addresses between them. If a Traffic Manager machine should fail, one of the other machines in its traffic IP group raises

its IP address. This, with the pools' failover system, gives full fault-tolerance for both front ends and back ends.

While a Traffic Manager is running, hostnames are re-resolved at regular intervals, usually of no more than 5 minutes. This provides an efficient re-resolving of nodes whenever the DNS changes.

Managing Your First Service

To begin managing your first service, you need to create a virtual server and a default pool. There are two ways to do this: using the **Manage a New Service** wizard or the **Configure** pages.

Browse to the address of the Admin Server home page. Log in with your username and password.

Note: If you chose not to provide a license key when you ran the `configure` script/**Initial Configuration** wizard, you can install one on the **System > Licenses** page. If no key is provided, the Traffic Manager will operate in Developer mode.

Using the Wizard to Create a Virtual Server and Pool

Click the "Wizards" drop-down menu and select the **Manage a New Service** wizard. Step through the instructions it gives.

1. Specify a name that you will use to identify the virtual server. Choose a protocol and port for the virtual server (e.g. HTTP, port 80).

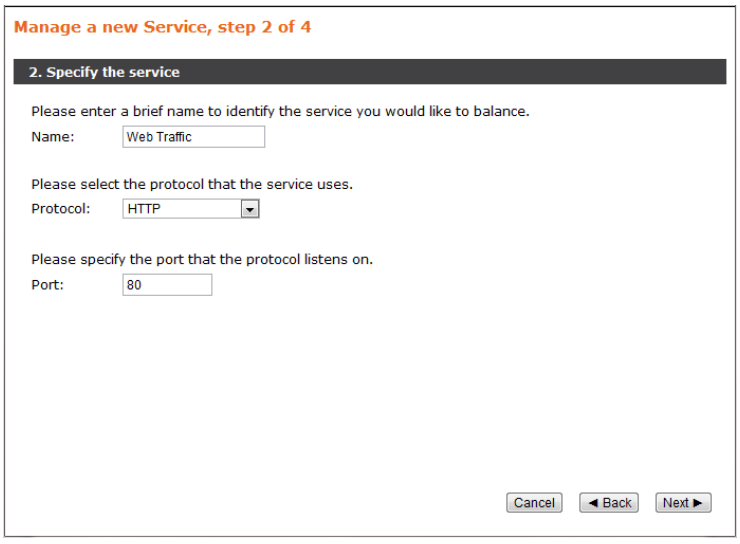


Fig 9. Three basic parameters to define the server: name, protocol, and port number

2. Create a list of back-end nodes, which will form the default pool for the virtual server. The nodes are identified by hostname and port. You can modify them later from the **Pools > Edit** page (see CHAPTER 5). You should ensure that you can serve content directly from the hostname-port combinations you specify.

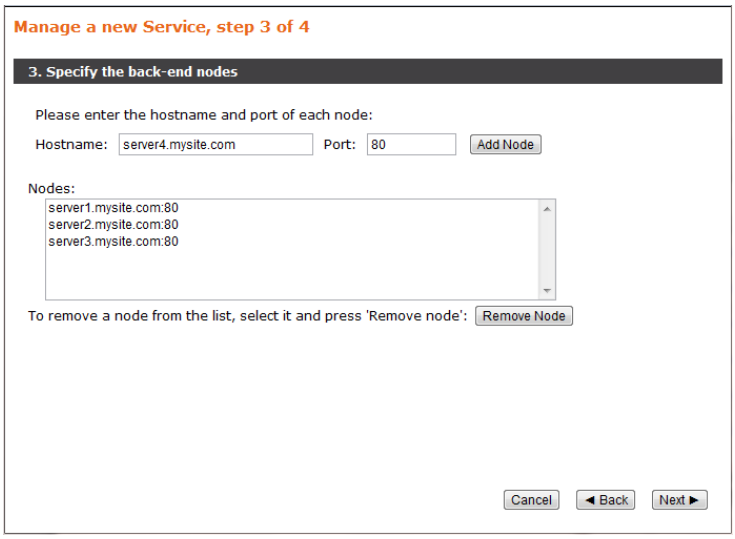


Fig 10. Creating nodes for the server

Note: If, for evaluation purposes, you are running a Traffic Manager on the same machine as your web server (or other test server), you need to make sure the two services are listening on different ports.

For example, suppose you are managing traffic to a Web server. The default HTTP port is 80; so you might have the Web server listening on port 8080 and the Traffic Manager listening on port 80. This means that entering the machine name in a browser (such as server1.mysite.com) will send traffic via the Traffic Manager.

3. Finally, review the settings you have chosen before clicking **Finish**.

4. You can now test your Traffic Manager setup by browsing to the machine and port you set up for your new service.

Note: Names for virtual servers, pools, error files, and so on, cannot begin with a period (.), an underscore (_), or a pound sign (#), and cannot end with a tilde (~).

Creating a Pool and Virtual Server Manually

Click the **Services** button on the top bar of the Admin interface. This takes you to the Services section, in which you can manage your virtual servers, pools, and Traffic Managers.

Creating a Pool

1. First you must create a pool of back-end servers. Click the **Pools** tab and fill in the details in the **Create a New Pool** section.

   You must choose a name for the pool, and enter a list of nodes (unless you enable Autoscaling, in which case no nodes are required). Each node should be listed in the form `server1.mysite.com:80`, where 80 is the port `server1` is listening on. The nodes should be separated by spaces:

   ```
   s1.mysite.com:80 s2.mysite.com:8080 s3.mysite.com:80
   ```

2. Click the **Create Pool** button. You will be taken to the **Pools > Edit** page for your new pool, where you can edit its basic and more advanced settings.

Creating a Virtual Server

1. Now click the **Virtual Servers** tab to create a new virtual server. You must specify a name, a protocol, and the port the virtual server is to listen on. If you are installing the Traffic Manager and your server software on the same machine, note the comment about port numbers in the previous section.

   You must select a default pool for the virtual server. Requests will be assigned to this pool unless a rule specifies another pool or action.

2. Click the **Create Virtual Server** button. You are taken to the **Virtual Servers > Edit** page for your new virtual server, where you can edit basic and advanced settings. These include the IP addresses (specified by domain name or address) the virtual server listens on.

Creating a Cluster

The Traffic Manager is often deployed as clusters of two or more instances for fault tolerance (see CHAPTER 6) and management reasons. All of the Traffic Managers in a cluster share the same service configuration and are managed as a single entity.

Note: System-specific settings, however, such as network configuration remain unique to each cluster member and must be managed individually.

You can join Traffic Manager systems together to form a cluster using one of the following methods:

- To create a new Traffic Manager cluster from scratch, choose one Traffic Manager as the first cluster member. Then, log in to the admin UI on each of your other Traffic Managers in turn, and use the **Join a cluster** wizard to join with the first Traffic Manager.

- To join an existing Traffic Manager cluster, login to the Admin UI on each new instance and use the **Join a cluster** wizard to join each of these with the existing cluster.

Note: In a cluster, all Traffic Managers are considered equal. You can access the Admin UI on any of your Traffic Managers, and the configuration changes you make are automatically replicated across the cluster. All of the Traffic Managers function together to provide fault tolerance and simplified management.

Joining a Cluster

Login to the Admin UI on one of your Traffic Managers and select **Join a cluster** from the "Wizards:" drop down box in the tool bar.

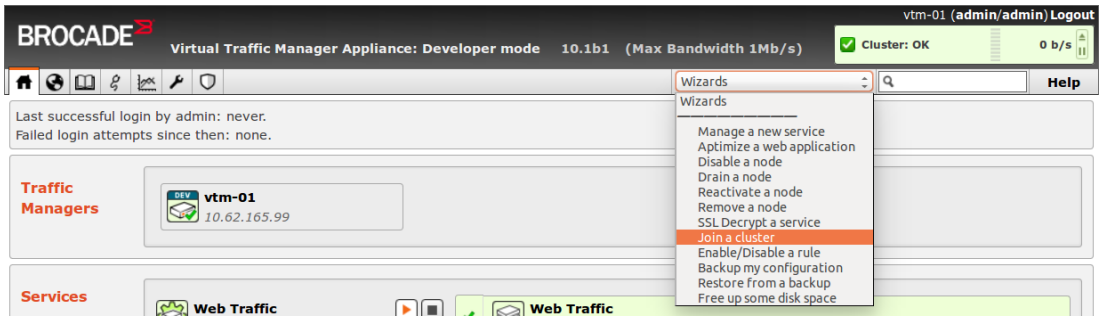


Fig 11. Creating a cluster via the Wizard

The Cluster Joining wizard starts in a separate pop up window.

Cluster Joining wizard, step 1 of 5
1. Getting Started
This wizard joins your current traffic manager to an existing cluster so that it can share the cluster's configuration and traffic.
Would you like to select an existing cluster from a list of available clusters on your network, or enter the Administration Server address and port of a specfic traffic manager to join?
Select existing cluster
Manually specify host/port
Cancel
◄ Back
Next ►

Fig 12. Step 1 of the Cluster Joining wizard

To instruct the Traffic Manager to automatically scan the network for contactable Traffic Managers, click "select existing cluster". Alternatively, to enter a specific hostname and port you want to join, click "Manually specify host/port". Click **Next** to continue.

The next step reflects the choice you make in step 1. If you clicked "select existing cluster", the Traffic Manager presents a list of discovered Traffic Manager instances and clusters.

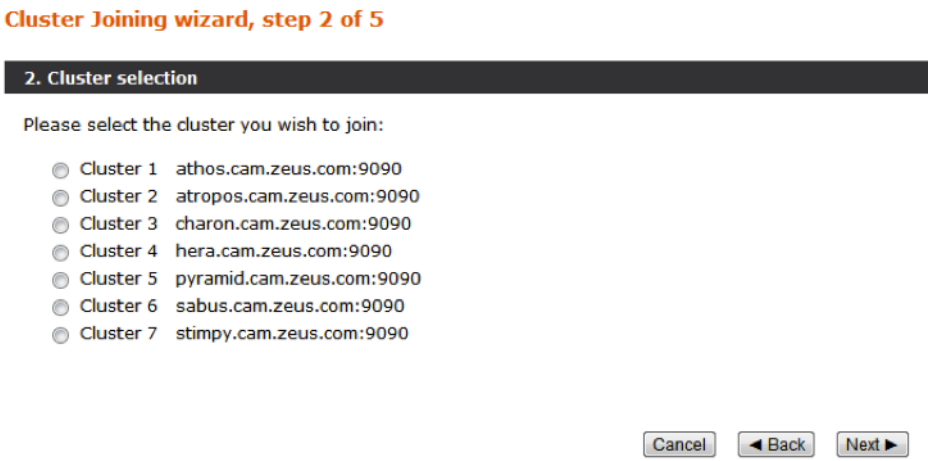


Fig 13. Selecting an existing Traffic Manager cluster to join

If you clicked "Manually specify host/port", enter your hostname and port number in the boxes provided.

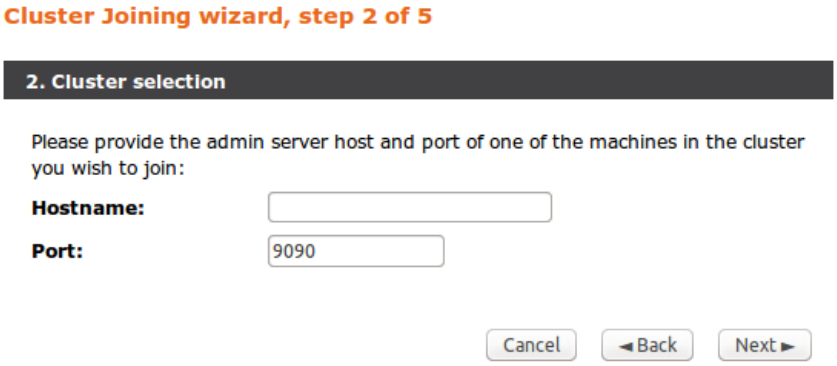

Cluster Joining wizard, step 2 of 5
2. Cluster selection
Please provide the admin server host and port of one of the machines in the cluster you wish to join:
Hostname:
Port:
9090
Cancel
◄ Back
Next ►

Fig 14. Enter a specific hostname and port number to join

Click **Next** to continue.

To connect to the specified instance or cluster, you must verify the identity of the Traffic Managers within the cluster, and provide the administration credentials used by the cluster.

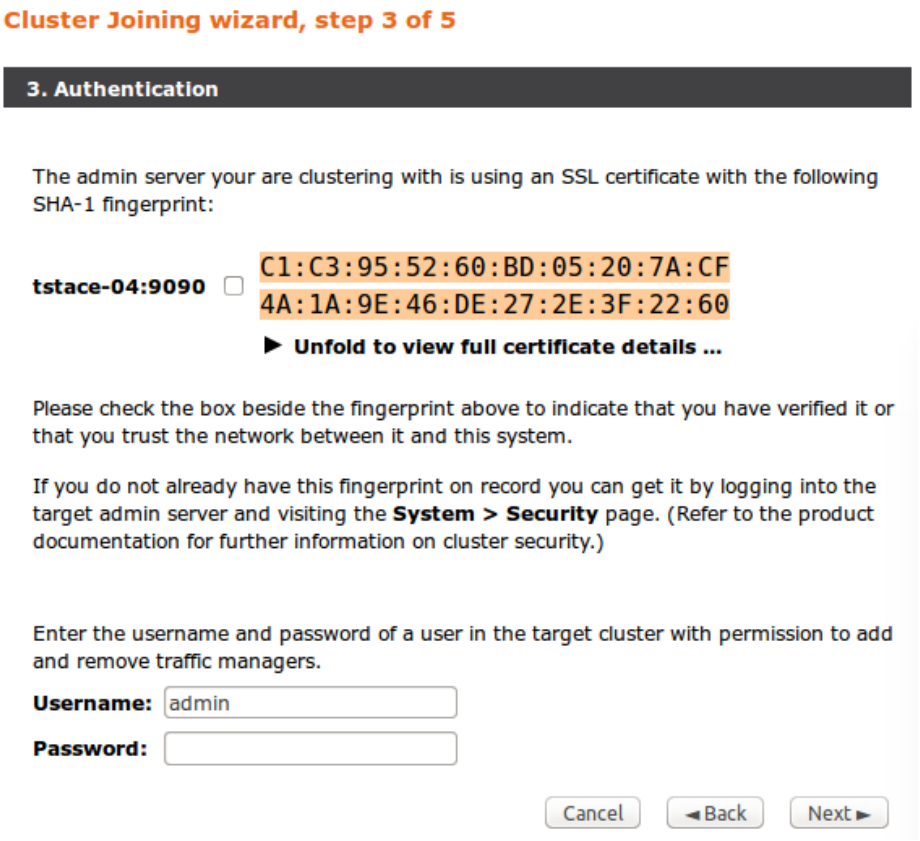

Cluster Joining wizard, step 3 of 5
3. Authentication
The admin server your are clustering with is using an SSL certificate with the following SHA-1 fingerprint:
tstace-04:9090
C1:C3:95:52:60:BD:05:20:7A:CF
4A:1A:9E:46:DE:27:2E:3F:22:60
Unfold to view full certificate details ...
Please check the box beside the fingerprint above to indicate that you have verified it or that you trust the network between it and this system.
If you do not already have this fingerprint on record you can get it by logging into the target admin server and visiting the System > Security page. (Refer to the product documentation for further information on cluster security.)
Enter the username and password of a user in the target cluster with permission to add and remove traffic managers.
Username:
admin
Password:
Cancel
◄Back
Next►

Fig 15. Authenticating the selected cluster

Brocade recommends that you verify the identity of each Traffic Manager in the cluster you want to join. To verify a Traffic Manager's identity, check the displayed SHA-1 fingerprint against the fingerprint shown in the target Traffic Manager's Admin UI, in **System > Security**.

Tick the checkbox next to each Traffic Manager hostname to confirm you trust it's identity, and then enter the cluster admin username and password. Click **Next** to continue.

Check your settings and click **Finish** to join the cluster. The Traffic Manager software reconfigures itself and presents a new home page showing all connected Traffic Manager instances in the Traffic Managers list.

To add further Traffic Managers to the cluster, run the **Join a cluster** wizard on the Admin UI of each Traffic Manager you want to add.

If you want to join a Traffic Manager to an existing cluster, but the cluster does not appear in the list in the wizard, check your network configuration and cabling and ensure that the network permits multicast and broadcast packets.

Joining Clusters with Traffic IP Groups

If the cluster already has one or more Traffic IP groups configured, the wizard can add the new Traffic Manager to these Traffic IP groups so that it starts handling traffic immediately.

However, this is likely to result in a number of connections being dropped at the instant the new Traffic Manager is added to the Traffic IP group, because allocations of traffic need to be transferred to the new Traffic Manager. In this case, you can select to add the new Traffic Manager as a "passive" member of the Traffic IP group. It does not accept any traffic until another member of the group fails.

CHAPTER 4 Virtual Servers

A virtual server manages all the traffic for a specified port and protocol. It can apply rules to decide which pool should handle a request, and can decrypt SSL traffic if required.

The Traffic Manager supports a wide range of protocols. Specialized handlers are provided for complex protocols such as HTTP, RTSP and SIP, and the software can load-balance TCP connections and UDP sessions using the 'Generic' protocol versions. Details about the protocol support available can be found in CHAPTER 11.

To modify settings for a virtual server, click the **Services** button and then the **Virtual Servers** tab. Your virtual servers are listed. You can create a new virtual server here, or click the name of an existing one to access the **Virtual Servers > Edit** page for that virtual server.

How Many Virtual Servers?

Generally, you should plan to run one virtual server for each distinct service you are running. That is, each TCP or UDP port you are accepting traffic on.

For example, you should run one virtual server for HTTP, irrespective of how many distinct Web sites you are running on port 80. This differs from the way you might configure a Web server, where you configure one virtual server for each distinct Web site. If you need subtly different traffic management configurations for each Web site, these can be implemented using TrafficScript to control how the traffic is managed.

A typical Traffic Manager installation will rarely have more than 5 or 10 virtual servers configured. If you anticipate having more than 100 virtual servers, you should plan to manage your Traffic Manager system using the programmatic Control API (see CHAPTER 25, "The Traffic Manager Control API") or REST API (see the *Brocade Virtual Traffic Manager: REST API Guide*) rather than the Web-based Administration Server.

Protocols

When you configure a virtual server, you need to specify the protocol of the traffic it is handling. The protocol value is the *internal protocol* used within the Traffic Manager to parse and interpret the traffic. Choose the most appropriate protocol setting for the traffic you are managing.

The 'Generic Server First', 'Generic Client First' and 'Generic Streaming' protocols are simple TCP protocol handlers that process TCP connections, forwarding data between clients and servers. The three protocols differ as follows:

- When the Traffic Manager receives a connection on a 'Generic Server First' virtual server, it immediately runs any TrafficScript™ rules, makes a load-balancing decision and connects to a back-end server. It then relays data bi-directionally between the client and the server.

  This selection is appropriate for protocols where the server application writes a 'handshake' or 'banner' message when the client application connects, before the client writes any data to the server.

- When the Traffic Manager receives a connection on a 'Generic Client First' virtual server, it does not process the connection until some data has been received from the client. The Traffic Manager then runs TrafficScript™ rules, makes the load-balancing decision, connects to the selected back-end server and writes the data previously received from the client. The Traffic Manager then relays data bi-directionally between the client and the server.

  This selection is appropriate for protocols where the server expects the client to initiate the dialog by writing data first.

- The 'Generic Streaming' protocol resembles 'Generic Server First', however is an appropriate choice where the virtual server is expected to handle non-request/response data originating from either the client or the back-end server. The Traffic Manager runs associated TrafficScript™ rules each time, makes a load-balancing decision and then relays data bi-directionally between the client and the server.

  This selection is appropriate for protocols where the server or client may initiate the dialog, but not automatically expect a response to any data received.

These generic protocols are the foundation that other TCP protocol handlers are built upon. The generic protocols allow you to inspect and rewrite request and response data and synchronize the communications between the client and the server, but only at a very low level.

The other TCP protocol handlers in the software are specialized for particular protocols – FTP, HTTP, etc. When you select a more specialized protocol such as HTTP:

- Additional protocol-specific options are made available in the configuration of the virtual server and pool, such as keepalive settings for HTTP or security settings for FTP.

- The Traffic Manager parses messages in that protocol and allows you to use additional TrafficScript functions to manage the request and response data more easily; helper functions to read and set Cookies in an HTTP transaction, for example.

You may use the appropriate basic 'Generic' protocol handler when you are processing a high-level protocol such as HTTP. If you do so, you will be unable to use the specialized handling for that protocol.

It also supports traffic over UDP; you can select either basic UDP (for a simple UDP request-response protocol such as DNS) or UDP streaming (where the server may send a limitless number of UDP packets in response to a request).

Please refer to CHAPTER 11 for more details about specific protocol support.

Note: If the traffic is encrypted using SSL and the Traffic Manager decrypts the traffic, then the protocol refers to the protocol of the decrypted traffic. For example, if the Traffic Manager is receiving HTTPS traffic on port 443 and uses SSL decryption to decrypt it, then the protocol in use within the Traffic Manager is 'HTTP'.

Applying Rules

A virtual server examines a request using TrafficScript rules to choose an action to take. Each rule examines the request, possibly modifies it, and performs a final action:

- Choose a pool to handle the request.
- Close the request.
- Log the transaction.
- Do nothing; the request is passed to the next rule in the list.

Each virtual server is associated with a ***default pool***. If no rule makes a positive routing decision, the request is assigned to this pool.

A rule can also selectively enable or disable features in the Traffic Manager for that specific connection. For example, a rule can specify that the request should use a particular session persistence class, or that the response should not be cached.

Rules are constructed using Brocade's TrafficScript language. This has the capability to inspect all aspects of the incoming request, from the source and destination port and IP to the type and actual content of the traffic. TrafficScript incorporates support for XPath (a language used to query XML documents), XSLT, and other XML-specific capabilities. These are often used by SOAP-based protocols employed by Web services, and enable complex data to be exchanged and understood automatically without user intervention.

Rules are applied in the order that you specify. You can apply rules to each incoming request, to each outgoing response, and at the completion of a transaction.

Create your rules in the rules catalog (see the **Catalogs > Rules** page of the Admin UI) and apply them through a virtual server. Any virtual server can use any rule from the catalog.

To add a rule to a virtual server, go to the **Virtual Servers > Edit** page for that virtual server and click the **Rules** link. For request rules, use the "Request Rules" section. Likewise, for response and transaction completion rules, use the "Response Rules" and "Transaction Completion Rules" sections, respectively. Select a new rule to add from the drop-down box in the desired section and click **Add Rule**.

The rules for your virtual server are shown in a list, and are applied in the order shown. Move a rule up or down the list using the *grab handle* to the left of each rule bar. Using the mouse pointer, drag and drop the rule to the new position you want it to be in; the Traffic Manager reorders the list automatically. If you have only one rule, no grab handle is displayed, and you cannot drag rules between sections.

You can disable a rule in the list to temporarily stop it from being executed, and re-enable it to make it active again.

For non-HTTP protocols that conduct a lengthy dialog with many requests and responses in one connection, you can choose whether a rule should be run *once* (just against the first request) or against *every* request. CHAPTER 8 covers creating and applying rules in more detail. The TrafficScript language is documented in a separate guide, available from the Brocade Web site at http://www.brocade.com.

SSL Decryption

A virtual server can decrypt SSL traffic. This can be useful for two reasons:

1. After decryption, a rule can analyze the request's headers and contents to make an informed routing decision. Without decrypting the packets very little information is available.
2. Decrypting requests requires processing power. It may be more efficient if the Traffic Manager decrypts requests before passing them on to the nodes, reducing the load on the back-end servers.

If traffic is decrypted in order to apply rules, you may wish to re-encrypt it before sending it on to the back ends. Re-encryption is handled by the pools (see the *SSL Encryption* section in CHAPTER 4).

To set up a virtual server to decrypt SSL traffic, go to the **Virtual Servers > Edit** page for that virtual server and click **SSL Decryption**. You can choose whether to decrypt

traffic, and which certificate from the SSL Certificates Catalog to use. You can also configure the allowed cipher suites and the SSL/TLS versions for each virtual server.

You can also choose whether to request an SSL client certificate. These serve to identify the client, and you can use them to restrict access to only those individuals you choose.

The Traffic Manager can also check client certificates using OCSP (Online Certificate Status Protocol). OCSP is an alternative to Certificate Revocation Lists (CRLs) and allows the Traffic Manager to obtain the revocation status of a client certificate. Clients making TLS connections can request that the virtual server supply status information for the server's certificate as part of the TLS handshake. Enable *OCSP Stapling* to instruct the Traffic Manager to retrieve the necessary OCSP responses and include them in its handshake messages.

The Traffic Manager's SSL capability is described in detail in CHAPTER 13.

Decrypting SSL Pass-Through Traffic

Recall that the protocol value for a virtual server refers to the internal protocol that the Traffic Manager is managing, after performing transformations such as SSL decryption.

If the protocol value for the virtual server is set to "SSL", this indicates that the virtual server is just forwarding SSL traffic in SSL pass-through mode. If you want to configure SSL decryption, you must first change the protocol value to the correct value for the internal protocol (e.g. HTTP). In this case, your pools are probably sending traffic to nodes which expect SSL encrypted traffic, so you will also need to configure SSL encryption in the pools.

You can use the **SSL Decrypt a Service** wizard to configure an SSL pass-through service to decrypt traffic in the virtual server, and re-encrypt it in the pool. This wizard is described in the *SSL Decryption Wizard* section of CHAPTER 13.

Note that only some protocols support SSL decryption. SSL decryption is not available for UDP based protocols, or for protocols that cannot be automatically wrapped with SSL such as SIP.

Service Protection Classes

A ***service protection class*** is a group of settings you specify to protect your service against malicious attacks, such as Denial of Service (DoS) and Distributed Denial of Service (DDoS). You can create a service protection class in the Service Protection Catalog, and configure settings such as:

- Lists of banned and trusted IP addresses. Connections from these IP addresses are never allowed and always allowed, respectively.
- Limits on the number of connections from one machine or a group of machines.
- A limit on the connection rate from any one IP address.
- Restrictions on HTTP requests, such as whether they should be strictly RFC2396-compliant.

You can apply a service protection class to a virtual server by clicking the **Classes** link on the appropriate **Virtual Servers > Edit** page. In the Service Protection section, select a class from the list and click **Update**.

Service protection is covered in detail in CHAPTER 15.

Bandwidth Management Classes

A ***bandwidth management class*** defines a bandwidth limit that virtual servers can apply to data sent to clients. For example, if several large download connections were assigned a shared limit bandwidth class of 250Kbits, these connections could not consume more than this limit.

You can apply a bandwidth management class to a virtual server by clicking the **Classes** link on the appropriate **Virtual Servers > Edit** page. In the Bandwidth Management section, select a class from the list and click **Update**. This will have the effect of limiting all the traffic the virtual server manages to the defined value in the class, according to the limit sharing type specified (per-connection, per-machine, or cluster-wide).

Bandwidth Management is covered in detail in CHAPTER 16.

Note: Bandwidth Management is not available on all Traffic Manager configurations. If required, it can be obtained via a software or license key upgrade.

Service Level Monitoring Classes

Service Level Monitoring (SLM) classes are used to monitor the level of service (response time) that end users of the service are receiving. An SLM class defines a desired response time. It also defines percentage tolerance limits; if the percentage of requests that meet the desired response time falls below these limits, an alert or log message is raised.

You can apply a service level monitoring class to a virtual server by clicking the **Classes** link on the appropriate **Virtual Servers > Edit** page. In the Service Level

Monitoring section, select a class from the list and click **Update**. This will cause the virtual server to monitor and log the response times for all of the connections it manages, and raise log+ messages or alerts in the event of service level problems.

Service Level Monitoring is covered in detail in CHAPTER 18.

Note: Service Level Monitoring is not available on all Traffic Manager configurations. If required, it can be obtained via a software or license key upgrade.

HTTP Content Caching

A Traffic Manager HTTP virtual server can detect commonly requested HTTP resources, and remember their content if it does not change each time it is requested. This capability is called 'Content Caching'.

When the virtual server sees subsequent requests for the same resource, it can return the content for the resource directly from the local cache, rather than forwarding the request on to a (possibly overloaded) back-end server. This capability reduces the load on the back-end servers and improves the performance and capacity of your HTTP services.

Clicking the **Content Caching** link on the **Virtual Servers > Edit** page shows you the settings for content caching. You can enable the content caching capability, and specify how long various types of content are cached for. Content caching is only available for HTTP virtual servers, and for virtual servers accepting and decrypting HTTPS traffic.

Content Caching is described in detail in CHAPTER 19.

Note: HTTP Content Caching is not available on all Traffic Manager configurations. If required, it can be obtained via a software or license key upgrade.

Aptimizer Settings

Aptimizer is the Traffic Manager's Web content optimization technology. It automatically optimizes Web page markup and elements, such as HTML code, images, scripts and custom style sheet, so they load faster for end users. Reducing Web page load time is vital to improving end user experience.

Where possible, versions of these optimized elements are internally cached for an even quicker response to subsequent requests. These accelerated Web pages are faster to load and can also save on data traffic and server infrastructure, thanks to better bandwidth utilization.

You can enable Aptimizer for your services by clicking on the **Aptimizer settings** link on the **Virtual Servers > Edit** page. Note that this facility is only available for your HTTP virtual servers.

Aptimizer settings are covered in detail in CHAPTER 20.

Note: Aptimizer functionality is not available on all Traffic Manager configurations. If required, it can be obtained via a software or license key upgrade.

HTTP Content Compression

The Traffic Manager can compress an HTTP response when it sends it to the remote client. This can reduce your bandwidth usage, and speed up the delivery of large Web pages to clients with slow connections.

Not all browsers can receive compressed content; those which do specify this in the HTTP request headers. The Traffic Manager compresses content only for those browsers which are able to decompress it.

Some Web servers are also able to compress content, so it may be more efficient to spread this work across your Web servers instead of using the Traffic Manager for this purpose. However, enabling compression on both should not cause any problems.

Clicking the **Content Compression** link on the **Virtual Servers > Edit** page shows you settings for content compression. You can choose whether to enable compression; which MIME file types to compress; and the size of documents that should be compressed.

If you offer large files for download from your site, it may be sensible to pre-compress them rather than have a server do this each time they are requested.

Controlling Content Compression

The Traffic Manager can compress HTTP responses if the request contains an '`Accept-Encoding: gzip`' header. The Traffic Manager checks the request before and after running TrafficScript rules; if the header is present in either case, the Traffic Manager will compress the content.

For example, you can configure a rule to remove the Accept-Encoding header from a request. In this case, the back-end server will never send a compressed response (because it never sees the header), but the Traffic Manager will compress the response from the back-end server if the remote client supports compression. This technique can be used to offload all compression tasks from the back-end server onto the Traffic Manager.

You can also control whether the Traffic Manager compresses content using the `http.compress.enable()` and `http.compress.disable()` TrafficScript functions.

Connection Analytics

The Traffic Manager has the ability to display a large amount of information about a particular connection. This can be a useful tool when tracing and debugging traffic handled by a virtual server. You can enable this feature under **Services > Virtual Servers > Edit > Connection Analytics**.

On this page you will find two main options, each with an additional option that is applicable only when enabled:

Configuration Option	Description
recent_conns!enabled	When enabled, connections handled by this virtual server are shown on the **Activity > Connections** page.
recent_conns!save_all	Select whether to show or hide, by default, all connections handled by this virtual server on the **Activity > Connections** page. You then override this behavior for individual connections using TrafficScript. Set to **No** to hide all connections by default, and then use the TrafficScript function `recentconns.include()` to selectively include connections in your rules. Set to **Yes** to show all connections by default, and then use the TrafficScript function `recentconns.exclude()` to selectively exclude connections in your rules. For more details about these functions, see the **TrafficScript Reference** in the Online Help or the **TrafficScript Guide** available from the Brocade Web site at http://www.brocade.com.

Configuration Option	Description
request_tracing!enabled	When enabled, the Traffic Manager collects data on internal events. This data includes the time that each request is received and when each rule is run.
request_tracing!trace_io	Set to **Yes** to enable the collection of data on individual read and write events for this connection.

Using Connection Analytics

With the earlier options enabled, use the **Activity > Connections** page to interrogate each individual request in detail. Click the magnifying glass icon next to each connection to view specific details. See the *Activity > Connections* section of CHAPTER 7 for more details.

Request Logging

Use this page to store detailed logs of the requests handled by your virtual server. You can write the logs to a local file on your Traffic Manager or to a syslog daemon running on a remote machine, according to the protocol described in RFC 3164. You can enable request logging for a virtual server by clicking the **Request Logging** link on the appropriate **Virtual Servers > Edit** page.

You can set the default logging behavior using the `log!save_all` setting. Set to **Yes** to instruct the Traffic Manager to log all connections by default. You can then use the TrafficScript function `requestlog.exclude()` inside a rule to exclude specific connections from the log. Set to **No** to instruct the Traffic Manager to not log connections by default. You can then use the TrafficScript function `requestlog.include()` in a rule to select specific connections to be included in the log. Click **Update** to apply your settings.

Request Logging to a File

You can specify a filename and log entry format, with details such as the IP addresses involved, the pool and node involved, and the bytes sent and received. You can also set HTTP-specific options such as the URL requested, the value of a specified header, and the HTTP method.

Note: Unlike other logs, request logs *are not* synchronized across your cluster. Each Traffic Manager maintains its own log files.

To log requests to a file, set `log!enabled` to **Yes** and specify the filename for the log. Click **Update** to apply your settings.

Logs can be automatically rotated so that each log does not grow too large. Use a `%{Time-String}t` macro in the filename; this macro expands to the current time or date.

For example, the macro `%{%Y%m%d}t` expands to a value like `20131020` (for October 20th, 2013). This will cause the log files to be automatically rotated at midnight as the value of the macro changes.

On virtual appliance and cloud instances of the Traffic Manager, log file naming and rotation is handled automatically:

- The log files are checked hourly.
- Any file that reaches 1 GB is archived.
- After the cumulative size of all current log files reaches 6 GB, the largest logs are archived by decreasing size until the cumulative total drops below 6 GB.
- All remaining logs are archived daily.

After the log directory reaches 85% of capacity, the Traffic Manager starts to delete the oldest files from the largest directories.

Viewing Request Logs

You can view request logs in real time by clicking the *View Request Logs* link. This redirects to the **Activity > View Logs** page, described in more detail in the *Activity > View Logs* section of CHAPTER 7.

Remote Request Logging

The Traffic Manager can log requests to a remote syslog server, rather than to a file on a local disk. All Traffic Managers in the same cluster log traffic to the same syslog server.

To configure remote logging, set `syslog!enabled` to **Yes**, specify an endpoint (IP address or host, and optional port), and select the log file format you want to use. Click **Update** to apply your settings.

Important: The syslog protocol uses UDP, which is not guaranteed to be reliable. When traffic levels are particularly high on a busy or *lossy* network, there is the danger of log records being dropped.

Controlling Request Logging

Requests can be selectively logged. The TrafficScript function `request.setLogEnabled()` can be used to enable or disable logging for an individual request; if it enables logging, the settings in the **Request Logging** page are used to determine how the request is then logged (local log file or remote syslog server).

Connection Management

From the **Virtual Servers > Edit** page, you can access advanced settings to manage connections between remote clients and your virtual server. These include the following:

Type	Description
HTTP settings	Keepalive settings, and whether the Traffic Manager should add an `X-Cluster-Client-Ip` header to the request.
Cookie settings	Specifies how the cookies are handled by the Traffic Manager.
RTSP specific settings	Allows user to define a specific range of ports to be used with RTSP (RTSP virtual servers only). Refer to the RTSP chapter for more detail.
SIP settings	Specify how the Traffic Manager handles SIP traffic (SIP virtual servers only). See the SIP chapter for more detail.
Location Header settings	Allows URL redirection in case a node replies with a 301 (moved permanently) or 302 (temporary redirect) status code, using an incorrect protocol, hostname or port (HTTP virtual servers only).
FTP settings	Security settings and port ranges for FTP data connections (FTP virtual servers only).
UDP settings	Timeouts, and the maximum number of response datagrams expected for one connection (UDP virtual servers only).

Type	Description
Timeout settings	Timeouts for new and established connections.
Connection Error Settings	Enable detailed logging of connection errors.
Memory limits	Maximum buffer sizes for data sent by the client and returned by the server.

Type	Description
Low-level settings	Enable "close_with_rst" to instruct the Traffic Manager to close connections from clients with a RST (reset) packet rather than a FIN (finish) packet. This avoids the TIME_WAIT state, which on rare occasions allows *wandering* duplicated packets to be safely ignored. It also ensures data has been fully received by the peer. Enable this option only if you fully understand the implications of TCP state transition, since TCP connections typically finish with a FIN packet to the peer and an acknowledgment FIN packet in reply. This mechanism allows the sender to be sure the peer has read all the queued data, which cannot happen if the connection is closed with a RST packet. On an unreliable network, data that has been transmitted can in some cases be lost in transit. The purpose of the TIME_WAIT state is to prevent wandering delayed packets from one TCP connection using address <A>:<port P> to address <B>:<port Q> from being accepted by a later TCP connection using the same addresses. In other words, if a duplicate packet from the first connection is delayed in the network and arrives at the second address when its sequence number is in the second connection's window, there is no way for the network stack in the kernel to determine that the delayed packet contains data from the first connection, potentially causing corruption. The kernel increments port numbers in use, so the possibility of this occurring is minimal but not zero. Enable proxy_close to send the client FIN to the back-end server and wait for a server response instead of closing the connection immediately. Enable so_nagle to improve the efficiency of the TCP connection by reducing the number of packets that have to be sent over the network.

Type	Description
Transparent Proxying	To enable transparent proxying for bound sockets on this virtual server, set "transparent" to Yes. In this configuration, a firewall rule can be used to send selected traffic to the virtual server without changing the destination address. In a TrafficScript request rule, you can determine what course of action to take by checking the request's destination IP address or port by using "request.getDestIp" or "request.getDestPort". You can then perform forward proxying to the original destination using "pool.select".

Settings specific to a certain protocol are only shown if the virtual server is managing that protocol.

You do not normally need to change any of these settings. The default values have been chosen to give good performance on a wide range of systems.

MIME Type Auto-Detection

The MIME Type is a component of an HTTP response that describes the type of data in the response – image, HTML page, etc – so that the client can know how to render or process it.

Some Web servers and Web applications do not correctly set the MIME type response header, and most client software is capable of detecting when this has occurred and auto-detecting the MIME type itself.

In certain circumstances, the Traffic Manager also needs to know the MIME type of a response. For example, the Traffic Manager will only attempt to compress certain types of HTTP response data; if the MIME type is missing or incorrectly identified, the Traffic Manager's content compression will not operate correctly.

In this situation, you can enable 'MIME Type auto-detection' in the **HTTP Specific Settings** part of the **Connection Management** configuration. The Traffic Manager will apply a range of heuristics to deduce the MIME type of the response data; auto-detection takes place when all of the following are true:

- There is a response body.
- The Content-Type header is missing, or it matches the value of **mime!default**.
- The response is not compressed.

The Traffic Manager will de-chunk data if necessary, if chunk-transfer encoding was used.

You should only configure MIME Type auto-detection when absolutely necessary, as it will reduce the performance of the Traffic Manager system.

Location Header Settings

This setting provides rewriting for the most usual URL redirection responses coming from a webserver, that is, 301 (object moved permanently) and 302 (object moved temporarily).

- location!regex - enter a regular expression to match the expected response from the node (a 301/302 redirection). You can use generic characters like (.*) to match parts of the expression.
- location!replace - enter a regular expression using $1 to $9 to redirect the response to a different object.
- location!rewrite - select the desired response in case the response from the node does not match the expression provided in location!regex.

For example, using the following values:

- location!regex= (.*)www.example.com/products/(.*)
- location!replace= $1products.example.com/$2

If the webserver returns a response such as the following:

```
HTTP/1.1 302 Moved Temporarily
Location: http://www.example.com/products/hacksaw.htm
```

The Traffic Manager rewrites it to:

```
HTTP/1.1 302 Moved Temporarily
Location: http://products.example.com/hacksaw.htm
```

Handling Errors

The **Connection Error Settings** section on the **Connection Management** page provides settings to enable logging of connection errors encountered by your virtual server.

Connection errors may occur for a variety of reasons:

- All of the nodes in the selected pool are unavailable.

- The connection with the back-end server timed out and could not be retried (see the *Retrying Failed Requests* section of CHAPTER 14).
- An SSL protocol error occurred in the server-side or client-side connection.
- The back-end server did not return a valid response (for example, a mal-formed HTTP response).

Client connection errors are likely to be due to the actions or behavior of clients, and are mostly not under the control of the Traffic Manager. Server connection errors relate to the Traffic Manager to back-end server connection and so are likely to be affected by, and ultimately can be corrected by, the Traffic Manager and your back-end server configurations.

Generally, connection errors are not logged, unless they relate to errors with the back-end servers that are detected by passive monitoring.

For debugging purposes, it can be very useful to log these errors:

- **log!client_connection_failures** - this setting enables verbose logging of client-connection errors to the error log, or another location if specified by the Event Handling settings in your Traffic Manager.
- **log!server_connection_errors** - this setting enabled verbose logging of server connection errors, and any internal faults in the Traffic Manager (such as a TrafficScript rule abort because the **max_instr** limit is exceeded).

For more information, refer to the troubleshooting documentation in CHAPTER 31 of this manual.

Returning a Custom Error Message

If the Traffic Manager is unable to obtain a valid response for a request, and the client connection is still alive, it will close the client connection or (for HTTP only) return the following error to the client:

```
Service Unavailable

The service is temporarily unavailable. Please try again
later.
```

This error message can be replaced, using the contents of a file uploaded to the `conf/extra` resource folder. Select the error message using the configuration setting **error_file**.

Files can be uploaded to the `conf/extra` resource folder using the **Catalog > Extra Files > Miscellaneous Files** configuration page.

The Traffic Manager is capable of processing HTTP headers within your error file, regardless of the actual file type, when you wish to set explicit directions on how the file should be handled. For example, you could set the `Content-Type` header if you wish to display a file of a particular type, such as a GIF or JPG image file. In this case, you would append the following lines to the top of your binary file:

```
Content-Type: image/gif

```

This capability can be extended to override the HTTP response code, as per the following example:

```
HTTP/1.1 200 OK

```

This would force the Traffic Manager to return the desired code in its response to the client.

Note: It is important to remember to include a blank line after any headers, before the actual content of the file.

Memory Limits for Connections

The Traffic Manager allows configurable limits on the amount of memory your virtual server can use for each connection:

- **max_client_buffer** – The limit on the amount of data the Traffic Manager receives from the client (to send to the server) that is buffered. The Traffic Manager responds differently when the limit is reached, depending on the state of the connection:
  - If the Traffic Manager is still reading in HTTP headers, it sends a 413 HTTP error code to the client and logs a warning.
  - If the Traffic Manager is still running request rules, it continues buffering and logs a warning.
  - If the Traffic Manager has reached the stage of streaming data to the server (request rule processing has finished), it applies flow control. In this scenario the client-side of the connection is paused, so the Traffic Manager does not read in any more data from the client until it has managed to push some of the buffered data to the server.
- **max_server_buffer** – The limit on the amount of data the Traffic Manager receives from the server (to send to the client) that is buffered. The Traffic

Manager responds differently when the limit is reached, depending on the state of the connection:

- If the Traffic Manager is still reading in response headers, it sends a 500 HTTP error code to the client and logs a warning.
- If the Traffic Manager is still running response rules, it continues buffering and logs a warning.
- If the Traffic Manager has commenced streaming the response to the client (response rule processing has completed), it applies flow control. In this scenario the server-side of the connection is paused, so the Traffic Manager does not read in any more data from the server until it has managed to push some of the buffered data to the client.

Note: It is possible for both limits to become active at the same time. If the Traffic Manager has reached the stage of streaming the request body to the server *and* the server starts to stream its response body before it has read the complete request body, and if both client and server are quick to send but slow to read, the Traffic Manager might activate both flow control mechanisms simultaneously.

CHAPTER 5 Pools

A pool is a logical group of back-end nodes. Each node is a combination of a server name or IP address, and port.

Examples of valid node entries are:

- server1.mysite.com:80
- 192.0.2.1:1234
- [2001:DB8:1b6c:6402:250:56ff:fea6:3f43]:4444

To specify a node as an IPv6 address, use the URI bracket notation format [<IPv6 address>]:<port>.

Note: The Traffic Manager does not support the use of IPv6 link local addresses for nodes.

A virtual server assigns requests to a pool, which load-balances them across its nodes. Each node in the pool must be able to receive requests through the port specified, using the virtual server's protocol.

As well as load balancing, each pool has its own settings for session persistence, SSL encryption of traffic, and autoscaling. These can all be edited via the configuration page for that pool.

To access the configuration page for a pool, click the **Services** button in the top menu bar and then click the **Pools** tab. All your pools are listed. You can create a new pool on this page, or click the name of an existing one to access the **Pools > Edit** page for that pool.

Load Balancing

The Traffic Manager offers a choice of load-balancing algorithms that distribute requests among the nodes in the pool. The algorithms are as follows:

Algorithm	Description
Round Robin	Connections are routed to each of the back-end servers in turn.
Weighted Round Robin	As for Round Robin, but with different proportions of traffic directed to each node. The weighting for each node must be specified using the entry boxes provided.

Algorithm	Description
Perceptive	Monitors the load and response times of each node, and predicts the best distribution of traffic. This optimizes response times and ensures that no one server is overloaded.
Least Connections	Chooses the back-end server which currently has the smallest number of connections.
Weighted Least Connections	Chooses the back-end server which currently has the smallest number of connections, scaled by the weight of each server. Weights can be specified in the entry boxes provided.
Fastest Response Time	Sends traffic to the back-end server currently giving the fastest response time.
Random Node	Chooses a back-end server at random.

Note: (Traffic Manager multi-site mode only) Node weightings cannot be switched between single and multiple location-based input like other areas of the Traffic Manager UI. Instead, if you have chosen to configure your nodes by location, the associated weightings will automatically be displayed by the locations used.

Which Load Balancing Method Is Best?

'**Least Connections**' is generally the best load balancing algorithm for homogeneous traffic, where every request puts the same load on the back-end server and where every back-end server is the same performance. The majority of HTTP services fall into this situation. Even if some requests generate more load than others (for example, a database lookup compared to an image retrieval), the 'least connections' method will evenly distribute requests across the machines and if there are sufficient requests of each type, the load will be very effectively shared. Weighted Least Connections is a refinement which can be used when the servers have different capacities; servers with larger weights will receive more connections in proportion to their weights.

Least Connections is not appropriate when individual high-load requests cause significant slowdowns, and these requests are infrequent. Neither is it appropriate when the different servers have different capacities. The '**Fastest Response Time**' algorithm will send requests to the server that is performing best (responding most quickly), but it is a reactive algorithm (it only notices slowdowns after the event) so it can often overload a fast server and create a choppy performance profile.

'Perceptive' is designed to take the best features of both 'Least Connections' and 'Fastest Response'. It adapts according to the nature of the traffic and the performance of the servers; it will lean towards 'least connections' when traffic is homogeneous, and 'fastest response time' when the loads are very variable. It uses a combination of the number of current connections and recent response times to trend and predict the performance of each server.

Under this algorithm, traffic is introduced to a new server (or a server that has returned from a failed state) gently, and is progressively ramped up to full operability. When a new server is added to a pool, the algorithm tries it with a single request, and if it receives a reply, gradually increases the number of requests it sends the new server until it is receiving the same proportion of the load as other equivalent nodes in the pool. This ramping is done in an adaptive way, dependent on the responsiveness of the server. So, for example, a new Web server serving a small quantity of static content will very quickly be ramped up to full speed, whereas a Java application server that compiles JSPs the first time they are used (and so is slow to respond to begin with) will be ramped up more slowly.

'Least Connections' is simpler and more deterministic than 'Perceptive', so should be used in preference when appropriate.

Caveats with Load Balancing Algorithms

Least Connections, Fastest Response Time and Perceptive can all have unexpected behavior at very low traffic levels. Least Connections will not distribute requests if you only ever subject it to one connection at a time; Perceptive and Fastest Response Time will tend to favor nodes with known good response times and will ignore nodes that are untested.

Load balancing metrics are not shared between Traffic Managers in a cluster. For example, if two Traffic Managers use the Round Robin algorithm to distribute requests, they will each progress through the nodes in turn, but independently.

Most load balancing metrics are shared between processes (CPU cores) when the traffic management software runs on a multi-core server. The one exception is response time information; this is not shared across cores.

If you are testing your back-end servers and want to be sure that traffic is directed to all of them (and want request distribution rather than load balancing), then use **'Round Robin'** or **'Random'** for your test traffic.

Locality Aware Request Distribution (LARD)

Perceptive, Least Connections and Fastest Response Time all use a technique called LARD (Locality Aware Request Distribution) to try and send the same request to the same back-end server. This technique takes advantage of server caching; if a server

returns a particular item of content, it is likely that it will be able to serve the same content quickly again because the content will be located in an internal cache (memory or disk).

When each algorithm makes its load balancing decision, it weights the decision with information as to which node processed the same request previously:

- 'Least Connections' will choose the favored node if there are several candidates for the node with least connections, and the favored node is one of them.
- Perceptive and Response Time algorithms give a light additional weight to the favored node in their internal selection.

Locality Aware Request Distribution is a lightweight way to advise the Traffic Manager how to route requests to back-end nodes. If you wish to mandate that requests for the same URL are sent to the same back-end server, you should use Universal Session Persistence (keyed by the URL) (see CHAPTER 12) to do this more forcefully. Alternatively, you can gain full control over request routing using TrafficScript™ and named node persistence, forward proxy or pool selection techniques.

Session Persistence

Session persistence is the process by which all requests from the same client session are sent to the same back-end server. It can be used for any TCP or UDP protocol.

A pool serving static Web content usually has no requirement for session persistence; each page or image for a particular client can be served from a different machine with no ill effects. Another pool, serving an online shopping site, may use session persistence to ensure that a user's requests are always directed to the node holding details about their shopping basket.

The Traffic Manager offers several methods to identify requests which belong to the same session. A variety of different cookies can be used; persistence can be based on a rule; or the client's IP address can be used to identify sessions. If incoming traffic is SSL-encrypted, the SSL session ID can be used.

You can choose what to do if a persistent session is lost. This might be due to invalid session data, or because the node handling it has failed. In this case you can choose to close the connection, have requests sent to a new node, or redirect the user to a specified URL such as an error page.

You can apply session persistence to a pool by clicking the **Session Persistence** link on the **Pools > Edit** page for that pool. Select a session persistence class and click the **Update** button.

Important: Care must be taken when using a persistence class that is already in use by another pool. You should only re-use a persistence class if the nodes in *this* pool match those in the pool already using the class. Session affinity cannot be guaranteed where a persistence class is in use by two or more pools with different nodes.

Session Persistence is described in more detail in CHAPTER 12.

Bandwidth Management

The Traffic Manager can apply limits on the bandwidth used by connections to the back-end nodes. These limits may be useful if the back-end nodes reside in a remote datacenter, and the bandwidth to the remote site needs to be managed.

You can apply bandwidth limits to a pool by clicking the **Bandwidth Management** link on the **Pools > Edit** page for that pool. Select a bandwidth management class and click the **Update** button.

Note: The bandwidth limit only applies to data sent from the Traffic Manager system to the remote server node. The Traffic Manager does not apply a limit on the bandwidth from the remote node back to the Traffic Manager. Note also that bandwidth management **does not** work with UDP-based protocols. Bandwidth Management is covered in detail in CHAPTER 16.

Note: Bandwidth Management is not available on all Traffic Manager configurations. If required, it can be obtained via a software or license key upgrade.

Health Monitoring

If the Traffic Manager sends a request to a node and receives no response, it assumes that node has failed. It stops sending it requests and balances traffic across the remainder of the pool.

The Traffic Manager's ***Health Monitors*** can run sophisticated health checks against back-end server nodes to determine whether they are operating correctly.

When a health monitor is assigned to a pool, it periodically checks each node in the pool; if it detects a certain number of failures, the Traffic Manager assumes that the node or pool is unavailable. These health tests are performed in addition to the Traffic Manager's connection tests when it attempts to send requests to and read responses from nodes.

The health monitors provide a range of tests, from simple tests, such as pinging each node, to more sophisticated tests that check that the appropriate port is open and the node is able to serve specified data, such as your home page. A number of pre-configured monitors are available to perform specific tasks (such as verify a POP3 login), and it is possible to create custom monitors to perform any test that is required.

Monitors fall into two categories: ***per-node*** and ***pool-wide***. A per-node monitor tests the health of each node in the pool. A pool-wide monitor performs tests on one machine which influences the health of the entire pool. For example, a mail server pool might keep its data on an NFS server, which each of your back-end servers accesses. A pool-wide monitor could test this server. If it fails, none of the back ends can retrieve the data so the whole pool is deemed to have failed.

On the **Pools > Edit** page, click the **Health Monitors** link to assign health monitors to a pool. When you have chosen your settings, click **Update** to apply them.

Health Monitors are described in more detail in CHAPTER 14.

SSL Encryption

You may wish to encrypt data before sending it from the Traffic Manager to the back ends.

On the **Pools > Edit** page, click the **SSL Settings** link to specify SSL encryption settings. When you have chosen your settings, click **Update** to apply them.

SSL configuration is covered in detail in CHAPTER 13.

Connection Management

By clicking the Connection Management link on the **Pools > Edit** page, you can configure settings to manage the connections between the Traffic Manager and the nodes. These settings include:

- Whether to force a connection from the Traffic Manager to the nodes to originate from a particular IP address.
- Whether to maintain HTTP keepalive connections to the nodes.
- The maximum number of times to establish a connection or wait for a response from a node.
- The maximum number of nodes to try before giving up and returning an error to the client.

- Whether to close connections from the Traffic Manager to the nodes with a RST packet rather than the default FIN packet. See "Connection Management" on page 72 for further information on the effect of this setting.

You do not normally need to change any of these settings. The default values have been chosen to give good performance on a wide range of systems.

Pool Connection Limiting

Introduction

All servers or applications use certain amounts of system resources to process a connection. These resources may include threads, file descriptors and an amount of memory. Each back-end server may be able to process a certain number of concurrent connections before handling more connections would result in sub-optimal performance. In extreme cases, the server can be seen to queue, drop or refuse new connections while the overloaded system attempts to keep up with demand.

In cases such as this, putting a hard limit on the number of concurrent connections that a server receives can keep it running at an optimal level while guaranteeing that it does not get overloaded. Pool connection limiting provides a way to limit the number of concurrent connections that can be made to a pool of nodes. Connection limiting can impact a number of other features of the Traffic Manager. So, it is important that you read the entirety of this section before using the connection limiter.

Pool Connection Limits

Connection limiting is configured in a pool, but is actually applied to each node within that pool. If you set a connection limit of 50 (for example), each node in that pool will receive a maximum of 50 concurrent connections. Use of the connection limiting feature assumes that all of the nodes within a pool are of the same specification and are providing the same application. If the same nodes are present in multiple pools, then the potential concurrent connections for that node will be the sum of the rate limits configured for each of those pools.

Clustered Connection Limiting

Connections to back-end machines are not shared amongst Traffic Managers in a cluster. In an active-passive scenario, all connections will be used by the active machine and none by the passive machine. In an active-active scenario, the connection limits will effectively double. In active-active-active they will triple, and so on.

Connection Queuing

When a node is chosen by a load balancing algorithm, if the number of connections to that node is currently at the specified limit, the connection will get stored in a queue. When the demand for connections continually exceeds the permitted connections for a long enough period of time, there is a risk (in the case of HTTP for example) that the response to the browser will time out. In order to help mitigate this risk, the TrafficScript function connection.checkLimits() can be used to detect when a pool has started queuing connections. This information can be used (for example) to respond immediately to the client with a different page rather than simply letting the response time out. Please refer to the *Brocade Virtual Traffic Manager: TrafficScript Guide* for more details.

Considerations

Connection limiting is a complicated feature, and it interacts with several other features of the Traffic Manager. This section describes those interactions so that you can know what to expect when using these features together.

Queue Management and SIP

RFC 3261 - SIP: Session Initiation Protocol Section 17.2.1 defines specific behavior in regards to what must happen when the response time of a SIP server is going to take longer than 100ms. The Traffic Manager has been developed to adhere to this part of the SIP specification.

“The server transaction MUST generate a 100 (Trying) response unless it knows that the TU will generate a provisional or final response within 200 ms, in which case it MAY generate a 100 (Trying) response.” [http://www.faqs.org/rfcs/rfc3261.html]

Connection Limiting and Request Rate Shaping

The Traffic Manager's request rate shaping feature uses a queue mechanism that is different from the one used by the connection limiting feature. Because a connection limit may cause a request to get queued while being subjected to connection limiting after being dequeued from a rate shaping class, it is possible for the rate of request getting sent to a node to vary from what was specified by the rate shaping class.

Connection Limiting and Service Level Monitoring

Service level monitoring can be affected by connection limiting. This is because it will be possible for a request to get queued by connection limiting. Time spent in the connection queue by a request will get added to the response time measured by an SLM class.

Enabling Pool Connection Limiting

To configure a connection limit for a pool, use the following steps:

1. Click Services > Pools > Edit > Connection Management.
2. Set the `max_connections_pernode` parameter to the desired number.
3. Scroll to the bottom of the page and click Update.

Connection Limit Options

You can configure the following additional options for connection limiting:

Configuration Option	Description
queue_timeout	Connections that are queued for more than queue_timeout seconds are discarded.
max_queue_size	Where the protocol for the associated virtual server is set to HTTP, the client is sent an internal server error (HTTP response code 500). The HTML page for this internal server error can be configured using the error_file parameter under the associated virtual server's "Connection Error Settings" configuration page.

Tracking Connection Limits

Because connection limiting involves queuing, it is possible that connections will time out. The current activity graph can be used to track the number of connections queued and the number of connections that time out while queued:

1. Navigate to Activity > Current Activity.
2. Click the Change data button.
3. Un-tick any boxes that are currently ticked.
4. Expand the Pools branch of the Values tree.
5. Tick the boxes that correspond to ConnsQueued and QueueTimeouts.
6. If you want to also check that the connection limiting is working, expand the Nodes branch of the Values tree.
7. Tick the CurrentConn box.
8. Scroll to the bottom of the page.

9. In the New Settings field, enter a name for your new chart (i.e. Connection Limits).

10. Click the radio button next to New Settings.

11. Click the Apply button.

Testing Connection Limits

The **zeusbench** utility can be used to check whether or not the connection limits you have configured are working properly. The zeusbench utility can be found in `$ZEUSHOME/admin/bin`. Running zeusbench in concurrency mode should be able to generate enough traffic to exhaust the free connections to your back-end nodes. If (for example) you have a pool of 2 nodes and you have configured a limit of 10 connections per node, the following zeusbench command should cause connections to be queued:

```
./zeusbench -c 100 -t 60 http://<your site>/index.html
```

zeusbench will attempt to open 100 connections to the virtual server you specify (by URL) and send as many requests down each one as fast as possible. This should result in a similar number of active connections being made to the pool under test. Keep in mind that the more nodes you have, the more concurrent connections you will need to specify when invoking the zeusbench utility.

Back-End Fault Tolerance

The Traffic Manager has a strong approach to fault tolerance. Pools can be configured in a number of ways to ensure that traffic is managed as efficiently as possible.

If a node in a pool fails, this is detected by the Traffic Manager's ***health monitors***. It can also be detected during load balancing, if the node does not respond to a request. No more requests are sent to that node, but are instead distributed across the other nodes in the pool. When the failed node recovers, it is brought back into service slowly until the Traffic Manager is satisfied that it can be relied upon.

A pool can be associated with a ***failure pool***. If every node in the original pool should fail, requests will be diverted to this failure pool.

To select a failure pool, click the **Services** button and go to the **Pools** page. Click the name of the pool you wish to edit. In the **Basic Settings** section you can select a failure pool from the drop-down list.

A typical failure pool for an HTTP service might consist of a single "sorry server". This can rewrite all requests it receives to request a simple Web page, which displays an "out of service" message.

Note: The Traffic Manager provides a direct way to serve a simple error page if a pool fails. Instructions to set this up, via the **Connection Management** page for that pool, are given in the *Connection Management* section above.

Priority Lists

Within a pool, you can set up a ***priority list***. This allows you to group the nodes in order of priority. You can specify the minimum number of machines you wish to receive traffic at any one time.

Suppose you have three servers, `primary1`, `primary2` and `primary3`, which you wish to use for normal traffic, and two off-site backup servers, `backup1` and `backup2`. You wish always to have at least two servers available.

The two backup servers could be set up as a failure pool for the three primary servers: however, the failure pool is only used if every node in the primary pool fails. If both `primary1` and `primary3` were to fail, only `primary2` would receive traffic.

Instead, you can set up a grouped priority list. This has two priority groups: {`primary1`, `primary2`, `primary3`} and, below it, {`backup1`, `backup2`}. You specify that a minimum of two nodes must be available at any one time.

With all the servers running, all three primary servers are in use; the backup machines do not receive traffic.

If `primary1` fails, there are still two nodes in the higher group available to take requests; these two nodes handle all the traffic.

If `primary3` now fails, only one node is left in the higher group. You have specified that two servers must be available. The Traffic Manager starts to send requests to *both* machines in the backup group: requests are now being handled by `primary2`, `backup1` and `backup2`.

No more priority levels are available. In the event that more nodes fail, traffic is balanced across the remaining nodes. If every node fails traffic will be passed to the pool's failure pool, if it has one.

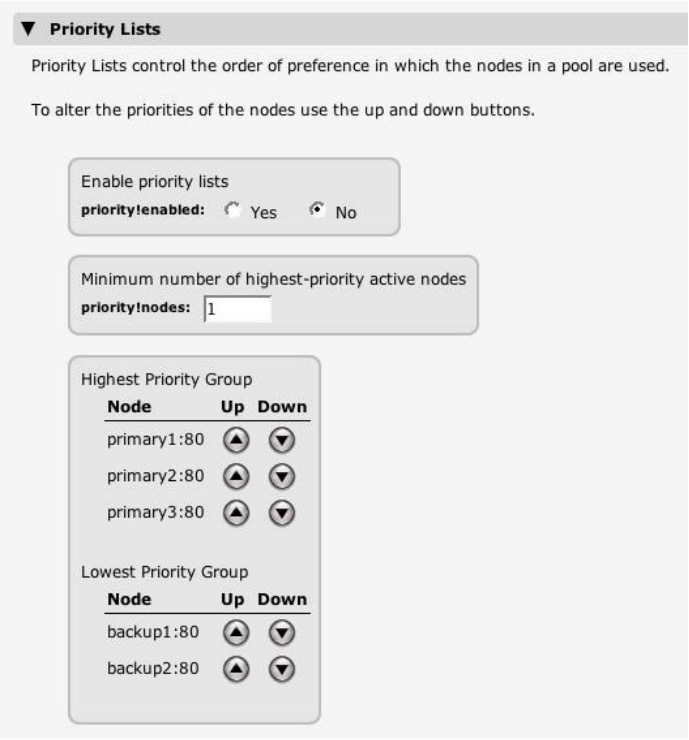


Fig 16. Example of priority group configuration

To set up priority groups for a pool, edit that pool via the **Pools > Edit** page. Click **Load Balancing**, and unfold the **Priority Lists** section.

The nodes in the pool are shown in a list, marked **Highest Priority Group**. To enable priority lists for this pool, click the **Yes** radio button and choose the minimum number of nodes you wish always to be available. Then use the up and down arrows beside each node to create other groups above or below the original one.

Note: The nodes in the lower priority groups are not used, until the number of nodes available in the highest group falls below the minimum you specified. Then *all* the nodes in the next group down are brought into service. The healthy servers from these top two groups are used, until enough of them fail that fewer than your specified minimum are available; at this point the *whole* of the next group is brought into use. This continues until all the priority groups are being used.

If every node in the pool should fail, requests will be directed to its failure pool, if one is configured.

Draining and Disabling Nodes

You may occasionally wish to remove one of your back-end servers from your load-balanced cluster. This could be permanent, or a temporary measure to allow for maintenance or upgrade.

You can choose to either ***disable*** or ***drain*** that node so that the Traffic Manager will not send it any new requests:

Disable a node: This operation stops the Traffic Manager sending any more connections to the node. Existing connections will complete as normal. The Traffic Manager stops monitoring the node.

Drain a node: This operation stops the Traffic Manager sending any more new connections to the node, but honors established sessions. If the Traffic Manager receives a request and session persistence requires that the node is used, the Traffic Manager will use it. Please refer to the *Draining Connections* section of CHAPTER 12 for more details. The Traffic Manager actively monitors the node.

You can disable or drain nodes via the **Pools > Edit** page. Edit a pool containing the node, change the status to **Disabled** or **Draining**, and click the **Update** button.

Note: This will only drain or disable the node for that particular pool. Any other pools using it will continue to send it traffic.

Disabling a Node

You should 'disable' a node if you plan to take it out of service temporarily, and you are not concerned about any sessions established with that node. If the node stops working (for example, because you shut down the server), the Traffic Manager will not report an error.

The advantage of disabling a node rather than removing it from the configuration is that a disabled node can easily be re-instated once it is ready to be used.

Draining a Node

You should 'drain' a node if you are concerned about sessions, but bear in mind that if the node stops working, the Traffic Manager will report an error. If you plan to shut the node down, you should drain it first, waiting until all sessions have completed, then disable it.

To find out whether the node is still handling connections, click the **Activity** button and go to the **Draining Nodes** tab. This page shows the number of connections the node is still handling, and the time since the last connection. When the node's existing connections have expired, you can disable it, or use the **Remove a Node** wizard to remove it cleanly from your cluster.

Using the Drain a Node and Disable a Node Wizards

To simplify the processes of draining or disabling nodes across a range of your services, the Traffic Manager provides **Drain a Node** and **Disable a Node** wizards. These can be accessed from the top menu "Wizards" bar.

You specify a node to drain/disable (depending on the wizard chosen), and choose whether to apply this for all services or just those you specify. You can restore the node to *active* status later using the **Reactivate a Node** wizard.

Autoscaling

Note: Autoscaling is a license key controlled feature and is not available on all Traffic Manager variants. See the Brocade Web site for details about supported configurations.

Introduction

The **application autoscaling** option monitors the performance of a service running on a supported virtual or cloud platform. When the performance falls outside the desired service level, the Traffic Manager can then initiate an autoscaling action, requesting that the platform deploys additional instances of the service. The Traffic Manager will automatically load balance traffic to the new instances as soon as they are available. The Autoscaling feature consists of a monitoring and decision engine, and a collection of driver scripts that interface with the relevant platform.

Autoscaling is a property of a pool. If enabled, you do not need to provide a specific list of nodes in the pool configuration. Instead, if performance starts to degrade, additional nodes can be requisitioned automatically to provide the extra capacity required. Conversely, a pool can be scaled back to free up additional nodes when they are not required. Hence this feature can be used to dynamically react to both short bursts of traffic or long-term increases in load.

A built-in service monitor is used to determine when a pool needs to be auto-scaled up or down. The service level (response time) delivered by a pool is monitored closely. If the response time falls outside the desired level, then Autoscaling will add or remove nodes from the pool to increase or reduce resource in order to meet the service level at the lowest cost.

How It Works

As mentioned above, the autoscaling mechanism consists of a *Decision Engine* and a collection of platform-dependent *Driver* scripts.

The Decision Engine

This monitors the response time from the pool, and provides scale-up/scale-down thresholds. Other parameters control the minimum and maximum number of nodes in a pool, and the length of time the Traffic Manager will wait for the response time to stabilize once a scale-up or scale-down is completed.

For example, you may wish to maintain an SLA of 250ms. You can instruct the Traffic Manager to scale up (add nodes) if less than 50% of transactions are completed within this SLA, up to a maximum of 10 back-end nodes. Alternatively, it should scale-down (remove nodes) progressively to a minimum of 1 node if more than 95% of transactions are completed.

Note: You can manually provision nodes by editing the max-nodes and min-nodes settings in the pool. If the Traffic Manager notices that there is a mismatch between the max/min and the actual number of nodes active, then it will initiate a series of scale-up or scale-down actions.

The Cloud API Driver

The Traffic Manager includes API driver scripts for Amazon EC2, Rackspace and VMware vSphere cloud environments. Before you can create an autoscaling pool, you must first create a set of *cloud credentials* pertaining to the cloud API you wish to use. These credentials contain the information required to allow a Traffic Manager to communicate with the aforementioned cloud providers. The precise credentials used will depend on the cloud provider that you specify. See the *Cloud Credentials* section of CHAPTER 7 for details.

The decision engine initiates a scale-up or scale-down action by invoking the driver with the configured credentials and parameters. The driver instructs the virtualization layer to deploy or terminate a virtual machine. Once the action is complete, the driver returns the new list of nodes in the pool and the decision engine updates the pool configuration.

Configuration

Important: The mechanism employed to implement autoscaling in a pool involves allowing the Traffic Manager to automatically make modifications to the pool configuration file (e.g. to add or remove nodes as they are required). Where such a pool is in active use, there is a small possibility that this process of automatic updates could interfere with configuration changes made by the user through the Admin UI or SOAP interface, if they occur simultaneously. Therefore, it is recommended that users double-check that their updates have taken effect by refreshing the pool page after making a change.

Once you have created a set of credentials for your chosen cloud API, you can proceed to configure the autoscaling properties of your selected pool. Click the **Autoscaling** link on the **Pools > Edit** page to display the available configuration options:

Basic Settings

Setting	Description
autoscale!enabled	Enables or disables Autoscaling. When this is enabled, nodes will be added and removed from the pool using the mechanism specified by "autoscale!external".
autoscale!external	Some cloud providers have their own mechanism for performing autoscaling. In this case, a cloud provider would monitor the performance for a group of machines (those being the machines in the pool) and will add and remove machines as needed. The cloud provider will indicate to the Traffic Manager that it has added or removed a machine using its API. When the Traffic Manager has been informed of the change, it will add or remove nodes from the autoscaling pool accordingly. If you intend to use your cloud provider's autoscaling mechanism, set this parameter to yes. If you intend to use the Traffic Manager's Autoscaling mechanism, set this to no.
Cloud Credentials	The set of credential required by a cloud's API. You can pre-configure sets of cloud credentials on the **Catalogs > Cloud Credentials** page.

The Traffic Manager uses the API type of the credentials you specify here to present a set of configuration keys applicable to the requirements of the cloud provider:

Amazon EC2 Cloud Settings

Setting	Description

Setting	Description
EC2 AMI	The unique identifier of the virtual machine image from which you create new node instances. The Traffic Manager adds each newly created node to the pool when it identifies the need to increase capacity. A Traffic Manager seeing instances in the cloud using this AMI assumes that those instances belong to the corresponding pool. The unique identifier used here determines the operating system and the services running on the node. In an EC2 cloud, this parameter is called **ImageId**.
EC2 Machine Type	The identifier that indicates the size--in terms of resources--of the virtual machine required for an instance of a node that is to be added to a pool. In EC2, this parameter is called **InstanceType**. **Note**: Typically, more powerful machines are associated with higher costs.
Name Prefix	An optional prefix added to the name of new nodes created during an autoscaling event. A Traffic Manager seeing instances in the cloud starting with this name assumes those instances belong to the corresponding pool.
EC2 Security Group IDs	A security group acts as a firewall that controls the traffic for one or more node instances. When an instance is launched, EC2 can associate it with one or more security groups. To create this association, specify a list of security group IDs (do not use group names) in the *Extra Arguments* text box. For EC2-Classic deployments, you must use security groups created specifically for EC2-Classic. Similarly, for EC2-VPC, you must use security groups created specifically for your VPC. If you do not specify security groups here, EC2 applies the default security group.

Setting	Description
EC2-VPC Subnet IDs	A list of IDs of the VPC subnets where the new EC2-VPC instances are launched. Instances are evenly distributed among the subnets you specify in the *Extra Arguments* text box. To launch instances inside EC2-Classic, leave the list empty. To distribute auto-scaled EC2-VPC nodes evenly across different availability zones, Brocade recommends listing the subnets present in different availability zones.
Extra RunInstances Arguments	A comma-separated list of key-value arguments for the AWS API function RunInstances, used at node creation time. Refer to the AWS documentation for RunInstances to see the list of available arguments.

Rackspace Cloud Settings

Setting	Description

Setting	Description

Image ID	The unique internal identifier of the virtual machine image from which you create new node instances. The Traffic Manager adds each newly created node to the pool when it identifies the need to increase capacity. The unique identifier used here determines the operating system and the services running on the node. In a Rackspace cloud, this parameter is called **imageId**.
	
	Important: You must use the internal image identifier here.
	
	To get a list of the image identifiers available for a given set of cloud credentials, run the included `rackspace.pl` script manually from the command line:
	
	`$ZEUSHOME/zxtm/bin/rackspace.pl \`
	`listimageids \`
	`--cloudcreds=<credentials>`
	
	Replace `<credentials>` with your Rackspace cloud credentials.

Setting	Description

Flavor ID	The internal identifier that indicates the size--in terms of resources--of the virtual machine required for an instance of a node that is to be added to a pool. In Rackspace, this parameter is called **FlavorId**.
	
	Note: Typically, more powerful machines are associated with higher costs.
	
	To get a list of the size identifiers available for a given set of cloud credentials, run the included `rackspace.pl` script manually from the command line:
	
	`$ZEUSHOME/zxtm/bin/rackspace.pl \`
	`listsizeids \`
	`--cloudcreds=<credentials>`
	
	Replace `<credentials>` with your Rackspace cloud credentials.

Name Prefix	The optional name prefix you can apply to each cloud-hosted virtual machine created by an autoscaling event. A Traffic Manager seeing instances in the cloud starting with this name assumes those instances belong to the corresponding pool.

VMware vSphere settings

Setting	Description
VMware Template	The built-in vSphere driver script creates new Virtual Machines (VMs) by deploying them from existing VM templates. These templates are pre-configured VMs which are marked as a template upon which new VMs can be based. The path of the template on a vCenter server is usually in the form: `<DatacenterName>/vm/<TemplateName>`

Setting	Description
Name Prefix	New nodes will be created with this name prefix on the vCenter Server. If multiple pools are autoscaling on the same vSphere infrastructure, care must be taken to keep the name prefixes for each pool unique.
Data Center	All ESX hosts managed by the vCenter server are arranged in logical datacenters. Autoscaling is performed on the hosts or clusters contained within this datacenter.
Data Store	The data-store that will be used by the newly created virtual machines. If left blank, the default data-store used by the datacenter will be used (depending on the vCenter configuration). **If a data-store is specified here, you should ensure the VM template uses the same data-store.**
VMware ESX Cluster	The infrastructure on a vCenter server usually has a datacenter at the top level, which contains ESX hosts or a cluster of ESX hosts. If a cluster is configured with Dynamic Resource Scheduling (DRS), vCenter decides the hosts upon which new virtual machines will be placed. If no cluster is configured, this setting should contain the ESX hostname or IP where the new VMs are to be created.

Node Settings

Setting	Description

Setting	Description
autoscale!ipstouse	(IP addresses-to-use) Instances in cloud environments typically have at least two IP addresses: one on a private network for communication with other nodes in the same cloud, and one on a public network for communication with the Internet in general. Select "private" or "public" to define whether traffic should be sent to the new node's private address, or its public address. Instances inside Amazon EC2-VPC do not have a public IP address by default. Select "public" to assign one at creation time.
autoscale!port	This defines the port to use when adding nodes to the pool. This port corresponds to the service listening on the back-end machine that gets spawned in the cloud environment.
autoscale!min_nodes	This defines the minimum number of nodes that are allowed to exist in the autoscaling pool. Note that this number will also include nodes that are currently marked as failed. A node failure in these circumstances can lead to a degradation in response times, which in turn will result in a new node being auto-scaled in. This behavior is not guaranteed, however. An autoscaling pool that contains failed nodes will not auto-scale beyond the number specified by autoscale!max_nodes.
autoscale!max_nodes	This defines the maximum number of nodes that are allowed to exist in the autoscaling pool. This ensures that a pool cannot auto-scale to an enormous number by itself.

Scaling parameters

Setting	Description

Setting	Description
autoscale!response_time	The Traffic Manager monitors the response times (in milliseconds) of the nodes in an autoscaling pool. This is measured from the time the Traffic Manager starts initiating a connection to the node until it receives the first byte of the response. A response time of less than autoscale!response_time is considered conforming and longer times are considered non-conforming. The Traffic Manager continuously measures the percentage of conforming connections.
autoscale!scaledown_level	When the percentage of conforming responses rises above this value for at least autoscale!hysteresis seconds, the pool is scaled down (i.e., a node is removed from the pool).
autoscale!scaleup_level	When the percentage of conforming responses falls below this value for at least autoscale!hysteresis seconds, the pool is scaled up (i.e. a node is added to the pool).
autoscale!refractory	After a change to the pool size has been made (be it an increase or a decrease), the Traffic Manager will wait autoscale!refractory seconds before making another change. It will take some time for the new node to have its effect on the overall response times of the pool. So to prevent too many nodes from being created (or destroyed) at a time, you may want to increase this time period.
autoscale!hysteresis	The amount of time (in seconds) that an autoscaling condition must exist before the Traffic Manager instigates a change. This corresponds to (for example) the time that the pool's percentage of conforming connections must remain below autoscale!scaleup_level before a scale-up event occurs.

Setting	Description
autoscale!lastnode_idletime	The time (in seconds) for which the last node in an auto-scaled pool must have been idle before it is destroyed. **Note:** This is only relevant if autoscale!min_nodes is set to 0.

DNS-Derived Autoscaling

The Traffic Manager can use DNS records to automatically expand or shrink the number of nodes in a pool to meet the demands of the traffic it is handling. This mechanism is known as DNS-derived autoscaling.

If a DNS record for a hostname resolves to multiple IP addresses, and the number of addresses changes depending on the resources available, that change can be reflected in the nodes used in a pool. For example, backend.example.com might resolve initially to four IPv4 records, and then later resolve to six IPv4 records. The Traffic Manager updates the node list in the pool to include the new IP addresses.

Configure DNS-derived autoscaling using the following settings:

Setting	Description
dns_autoscale!enabled	To enable DNS-derived autoscaling for a pool, set this key to Yes. By default, pools are not auto-scaled.
dns_autoscale!hostnames	A list of DNS hostnames that the Traffic Manager periodically queries. Any IPv4 A or IPv6 AAAA records are added or removed to this pool's list of nodes.
dns_autoscale!port	The port number the Traffic Manager appends to every node added using DNS-derived autoscaling.

CHAPTER 6 Traffic IP Groups and Fault Tolerance

A cluster of Traffic Managers can distribute incoming network traffic between them, and transfer traffic shares from one to another if a Traffic Manager fails. Traffic distribution is configured by means of Traffic IP Groups.

The functionality referred to in this chapter might vary slightly depending on your product platform. For additional details and product specifications, see the Brocade Web site at http://www.brocade.com.

Fault Tolerance

Traffic IP Addresses and Traffic IP Groups

A **Traffic IP Address** is an IP address that must remain highly available. These Traffic IP addresses are assigned to groups, called **Traffic IP Groups**.

Each Traffic IP Group is managed by some of (or all of) the Traffic Managers in your cluster. They cooperate and share the traffic between them, ensuring that any traffic to the Traffic IP addresses is distributed between and managed by one of the Traffic Managers in the cluster.

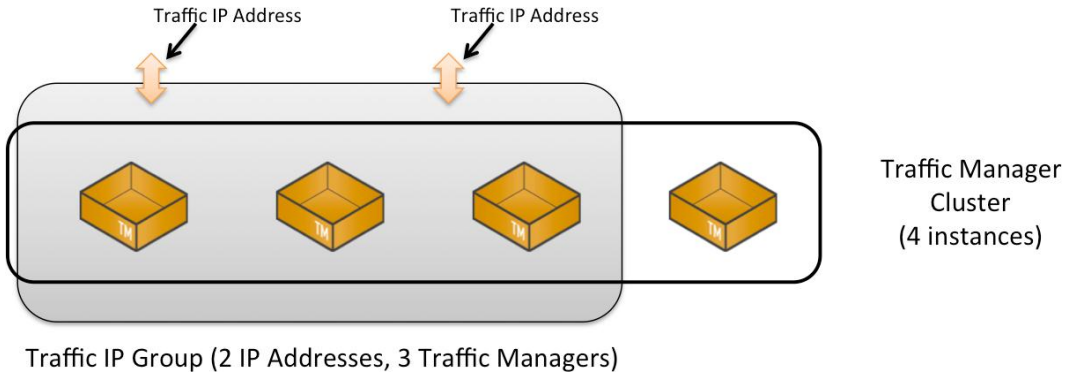

Traffic IP Address
Traffic IP Address
Traffic Manager Cluster (4 instances)
Traffic IP Group (2 IP Addresses, 3 Traffic Managers)

Fig 17. Sample Traffic IP address configuration

In the illustration above, a Traffic IP Group has been configured, spanning 3 of the 4 Traffic Managers in the cluster. This Traffic IP Group will manage the Traffic IP addresses, and the three Traffic Managers will ensure that those Traffic IP Addresses are available.

For example, a Web service might be published on IP address 34.56.78.90. You can ensure that the service is always available by adding that IP address to a Traffic IP group. The Traffic Manager cluster raises that IP address and manages all of the traffic to it. You would typically configure the DNS entry for your service to resolve to one or more Traffic IP addresses.

Distributing Traffic Within a Traffic IP Group

Traffic is shared between the Traffic Managers in your cluster using one of the following distribution policies:

- **Single-Hosted mode**: Each Traffic IP address is raised on one of the Traffic Managers in the group. If there are multiple IP addresses in the group, they are raised on different Traffic Managers, distributed as evenly as possible.
- **Multi-Hosted mode**: Each Traffic IP address is raised on all of the Traffic Managers in the group. Traffic to each IP address is evenly shared between all of the Traffic Managers.
- **Route Health Injection:** (not applicable to all product variants) Each RHI traffic IP group contains one active Traffic Manager, and optionally one passive Traffic Manager. All Traffic IP addresses in the group are raised privately (on loopback) by both participating Traffic Managers and dynamically advertised into the adjacent routing domain using OSPFv2 or BGP as the routing protocol. In response, routers direct all traffic to the active Traffic Manager.

Each Traffic Manager reports periodically to the others, and if one fails, the remaining Traffic Managers take over their share of traffic. In this way, services that depend on the Traffic IP addresses are always available.

For a discussion on example configurations of Traffic IP Groups, see "Example Configurations" on page 39. To understand how to create a Traffic Manager cluster, see "Creating a Cluster" on page 56.

Choosing Traffic IP Addresses

The Traffic Manager is typically attached to a front-end network for incoming traffic and a back-end network for back-end servers. At the IP level, these networks are described as IP subnets. For example, 192.0.2.0/24, 198.51.100.0/16, and so on.

For single-hosted and multi-hosted traffic IP groups, you must choose traffic IP addresses that are within these directly attached subnets. The Traffic Manager raises these IP addresses on the correct physical interface, so that routers and other devices on those networks can reach them directly.

For RHI traffic IP groups, you must choose traffic IP addresses that are not within these directly attached subnets. The Traffic Manager always raises RHI traffic IP addresses *privately* in their own /32 subnet on the Traffic Manager's internal loopback (lo) interface. Routers and other machines always send traffic to them *indirectly*, through the Traffic Manager's front-end IP address, according to routing information established using the OSPFv2 routing protocol.

The directly attached subnets are those you specify:

- When configuring a statically raised IP address on a particular interface.
- By adding an entry into the "Traffic IP Networks" table. For more information, see "Interface-to-Subnet Mapping (Traffic IP Networks)" on page 109.

Creating a Traffic IP Group

To create a traffic IP group, click **Services > Traffic IP Groups**. On this page, you can create new traffic IP groups and edit existing ones.

Enter a name and select the Traffic Managers you want to be members of the group. Enter the traffic IP addresses for the group in a list separated by spaces or commas, choose the IP distribution mode if necessary, and click **Create Traffic IP Group**.

Traffic Distribution

The IP distribution mode determines how Traffic Managers in the same Traffic IP Group share the traffic between them.

You can configure traffic IP groups to raise traffic IP addresses on one Traffic Manager at a time (single-hosted mode), or on all Traffic Managers in the group simultaneously (multi-hosted mode). On some product variants, you can also select Route Health Injection as the distribution mode.

Single-Hosted Mode

Each IP address in the group is only raised on one Traffic Manager at a time. If the group contains more than one IP address, they are dispersed across the participating Traffic Managers. When a Traffic Manager fails, its traffic IP addresses are moved to the other working Traffic Managers in the group.

Set the **keeptogether** option for a traffic IP group to prevent dispersion of the IP addresses in the group. In this case, all traffic IP addresses are raised on the same Traffic Manager. This is useful when using traffic IP groups with IP Transparency. For details, see "Using IP Transparency with a Cluster" on page 42.

Multi-Hosted Mode

Each IP address is raised on every machine at the same time. Incoming data for each IP is received by every Traffic Manager in the group, and each one takes responsibility for an equal portion of the incoming connections. If a machine fails, its portion is shared between the remaining active machines.

Multi-hosted IP addresses work by using multicast packets on your local network. The Traffic Manager advertises the traffic IP address using a multicast MAC address

on the local subnet. This informs switches and routers that the traffic to the IP should be sent to all of the Traffic Managers. The MAC address used is calculated by providing a separate multicast group (represented as an IP between 239.0.0.0 and 239.255.255.255). This multicast group should be unique on your network.

Each Traffic Manager receives all incoming connections and makes a calculation based on the source IP address of the connection to determine whether it should accept that connection or if it should silently discard the connection (because another Traffic Manager in the group will accept it).

By discriminating on the source IP address alone, this method arranges that the same client will be processed by the same Traffic Manager (providing the client does not change IP address). This has the advantage that any session persistence data and SSL session data stored in the Traffic Manager will be available.

However, this can result in uneven distribution of traffic between Traffic Managers, particularly if your clients come from a small set of source IP addresses. This uneven distribution may impact the performance of the load-balanced service. If this is of concern, set the **consider source port** setting to instruct the Traffic Managers to distribute traffic based on the source IP address and port; this will share traffic much more easily across the cluster.

Note that using the source port may interfere with some session persistence methods, as there can be a short delay as the session persistence data is shared across the cluster (see the *Using Session Persistence with Multi-Hosted Traffic IP Addresses* section of CHAPTER 12). Similarly, **consider source port** should not be used with any Traffic IP Addresses that are used by FTP virtual servers (see the *FTP* section of CHAPTER 11).

Note: If a Traffic Manager attempts to connect to a multi-hosted IP, the connection is always picked up by the local Traffic Manager itself.

Route Health Injection

The Traffic Manager uses front-end fault tolerance by announcing the Traffic IP addresses managed by the group out to the wider routing network. To use this method, the routers in your organization must be configured to use one of the following routing protocols:

- **OSPFv2**: Open Shortest Path First, version 2
- **BGP**: Border Gateway Protocol

You can only use RHI for IPv4 addresses; IPv6 is not supported.

Traffic IP Groups based on Route Health Injection (RHI) contain a maximum of two Traffic Managers, either with a single active Traffic Manager, or in a pair using an active-passive configuration.

When you select RHI as the distribution method, you additionally specify routing metrics for the active and passive Traffic Managers. When the Traffic Manager injects routes into the upstream routing network, these metrics determine the priority of the Traffic Manager instance within a multi-cluster (or multi-datacenter) scenario.

For more information, see "Route Health Injection" on page 45.

Passive Machines

If you mark a Traffic Manager as passive then it will not raise any of the IP addresses (single-hosted mode) or handle any load (multi-hosted mode) in the Traffic IP group unless one of the non-passive Traffic Managers has failed.

When you add a Traffic Manager to an existing Traffic IP Group, some of the traffic may be transferred to the new Traffic Manager. Unavoidably, this will result in some dropped connections at the instant of transfer. To deal with this situation, you can add a Traffic Manager as passive. In that case, it will not take any traffic unless one of its peers was to fail.

Disabling a Traffic IP Group

You can disable a Traffic IP Group by de-selecting the **enabled** checkbox. Use this option to disable a group temporarily, but to retain the configuration for future reinstatement.

When a group is disabled, none of the IP addresses in the group are raised. Route advertisements for RHI traffic IP groups are withdrawn.

Interface-to-Subnet Mapping (Traffic IP Networks)

Prior to version 7.0 of the Traffic Manager, there was a pre-requisite to hosting a Traffic IP Address of having an interface that is configured with an IP address in the same subnet. This pre-requisite provides a benefit because it allows a Traffic Manager to automatically figure out which devices can host a Traffic IP Address (based on the routing table and “health” of the interface). The drawback to this is that it means you end up with an IP address that does not actually receive any traffic, but still takes up space. This can be frustrating if the network you want to receive traffic in is small (say, a /29 for example). The interface-to-subnet mapping feature gives you a way to define an explicit mapping of subnets to interfaces. When in use, you do not need to configure an interface with an IP address in a particular subnet just to host a Traffic IP Address on it.

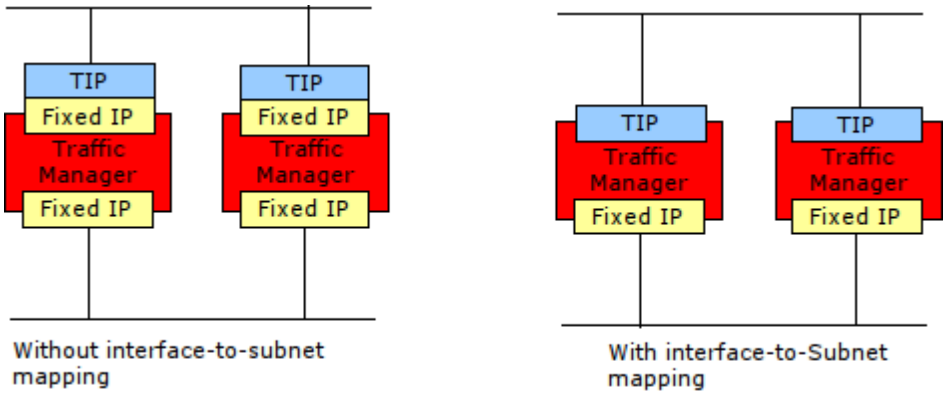


Fig 18. Interface to subnet mapping

To configure a Traffic IP Network

1. Navigate to the **Services > Traffic IP Groups** page.
2. In the **Traffic IP Networks** section, click the **Network Settings** link.
3. In the **Add Network** field, using **CIDR notation**, enter the IP subnet that you want to host IP addresses in. For example, 192.168.50.0/24.
4. From the **Default Interface** drop-down box, select the network interface that you want this subnet to be hosted on.
5. Click **Update**.

After configuring a Traffic IP Subnet on the desired interface, you will see a dialogue box appear that allows you to configure which interfaces should be used for the same subnet on the other Traffic Managers in the cluster.

Configuring Fault-Tolerance

The Traffic Managers in a cluster periodically check that they can communicate with the network using ICMP pings through each active network interface. They then broadcast a message describing their health (good or failed).

If one of the Traffic Managers in a cluster fails, the other Traffic Managers will take over any Traffic IP addresses that the failed Traffic Manager was managing.

Fault Tolerance Configuration Settings

You can configure fault tolerance using the settings in **System > Fault Tolerance > General**:

- **flipper!autofailback** - If a Traffic Manager that is hosting Traffic IP addresses fails, another Traffic Manager in the cluster raises the dropped addresses until the original Traffic Manager recovers.

  Set this to Yes to ensure that the Traffic IP addresses are automatically returned to the original Traffic Manager when it regains connectivity.

  Set this to No to force the original Traffic Manager into a *pending* mode upon recovery, in which case it waits to be instructed when it should raise the Traffic IP addresses.

  The Admin UI displays a warning when a Traffic Manager is in the pending state. Use the **Diagnose > Cluster Diagnosis** page or the **Services > Traffic IP Groups** page to fully reactivate a pending Traffic Manager.

  Note: A pending Traffic Manager is automatically reactivated if all other Traffic Managers in the cluster fail. In this case, the Traffic Manager also takes over all Traffic IP addresses it is responsible for. A Traffic Manager also reactivates automatically if it would not cause any disruption to the services already running.

- **flipper!monitor_interval**: The time interval, in milliseconds, that the Traffic Manager uses to periodically check if it can contact other devices on the network. The result is then announced to other Traffic Managers on a multicast address. Decrease the interval to ensure failures in the network are detected sooner, however this increases the workload on the system.

- **flipper!monitor_timeout**: The amount of time, in seconds, that the Traffic Manager should wait for a response when it has tried to contact one of the devices used to check connectivity. If a response is not received within this time, the Traffic Manager assumes it cannot contact the device. Decrease this value to allow the Traffic Manager to detect failures sooner, although the reliability of the system is reduced because slower connections might be incorrectly regarded as having failed.

- **flipper!heartbeat_method**: In a cluster, each Traffic Manager must send out periodic *heartbeat* messages to advise that all software and network services are running.

  Use Unicast to instruct the Traffic Manager to send directed UDP messages to each Traffic Manager in your cluster. Use this setting if the switches in your network are unable to handle the potentially more efficient multicast packet method. Unicast messages can also reach machines on separate subnets; for example, Traffic Managers in disparate data centers. Use **Communication Port:** to specify the network port the Traffic Manager must listen on for unicast messages. This port must be contactable from all other Traffic Managers in the cluster.

Use Multicast to instruct the traffic Manager to send these messages using multicast network packets, which should traverse all network ports on all switches that your Traffic Managers are connected to. Multicast messages only reach machines on the same subnet. Set the multicast address that all Traffic Managers should listen on using the **Multicast address and port:** field (by default, this is 239.100.1.1:9090).

- **flipper!use_bindip**: Should multicast messages be sent and received on the management interface alone.

  Set this to Yes if your Traffic Managers communicate with each other on a separate, dedicated network to that used by the main traffic. Note that, however, the reliability of the system could be compromised as a result.

  Set this to No (default) to instruct the Traffic Manager to send the multicast messages it uses to announce its connectivity out over every network interface. This increases the reliability of the system by ensuring that failure of one network interface does not cause other Traffic Managers in the cluster to assume that this machine has failed.

- **flipper!arp_count**: The number of ARP packets that the Traffic Manager sends when a network interface is raised. Although other devices on the network only requires one ARP request to acknowledge a new interface on a Traffic Manager, ARP packets can get lost or missed. To improve the reliability and speed of the broadcast, the Traffic Manager sends the number of ARP packets defined here when an interface is raised.

- **flipper!igmp_interval**: The time interval, in seconds, that the Traffic Manager uses to sends unsolicited IGMP Membership Report packets for Multi-Hosted Traffic IP groups. Use this interval to prevent switches and routers from forgetting the Traffic Manager's multicast subscription(s). To disable this feature, set the interval to 0.

- **flipper!verbose**: Should a Traffic Manager log all connectivity tests. Set this to Yes to output detailed information to the system log.

  Note: Brocade recommends only using this for diagnostic purposes.

- **flipper!frontend_check_addrs**: The IP addresses to use for front-end connectivity checking. If the Traffic Manager cannot successfully *ping* these addresses, it decides that its network connectivity is broken and performs fail-over to the other Traffic Managers in the cluster.

  If your Traffic Manager setup does not require external connectivity; for example, if it is part of an Intranet, set this field to blank to disable the external connectivity check.

Understanding Traffic Manager Fault Tolerance Checks

Each Traffic Manager checks that its network interfaces are operating correctly. It does this by:

- Periodically pinging the default gateway, to ensure that its front-end network interface is functioning.
- Periodically pinging each of the back-end nodes in the pools that are in use, to ensure that its back-end network interfaces are functioning.

The Traffic Manager concludes that it has failed if it cannot ping the default gateway, or if it cannot ping any of the nodes in any of the currently used pools.

Overriding Front-End Checks

You might need to override the front-end check if, for example, your gateway does not respond to ICMP pings. To override the front-end check, set **flipper!frontend_check_addrs** to an empty string.

Health Broadcasts

Each Traffic Manager regularly broadcasts the results of its local health checks, whether it is healthy or not.

By default, each machine broadcasts these heartbeat messages twice per second. You can configure this behavior using the **flipper!monitor_interval** setting.

You can choose to send broadcast messages by one of two methods, unicast or multicast. Select your preference with the **flipper!heartbeat_method** setting.

Important: If you use unicast, you must ensure that *State Sharing* is enabled. To do this, set **state_sync_time** on the **System > Global Settings** page to a non-zero value.

In many circumstances, unicast is appropriate. However, if your network setup means that all Traffic Managers can receive multicast messages, for example, if your Traffic Manager cluster is within a single network segment, use multicast instead. If, after trying both methods, you still encounter issues, contact your support provider for assistance.

Note: Brocade recommends that users of Hyper-V-based Traffic Managers use the default unicast method. When two or more instances are clustered together, using multicast can, in some circumstances, result in packet loss.

To debug heartbeat communication problems, Brocade recommends using the `tcpdump` program to capture all heartbeat messages on your network:

```
# tcpdump –i eth0 ip multicast
```

To capture only the heartbeat messages issued by your Traffic Managers, use:

```
# tcpdump -i eth0 dst host 239.100.1.1 and dst port 9090
```

Determining the Health of a Cluster

Each Traffic Manager listens for the health messages from all of the other machines in the cluster. A Traffic Manager concludes that one of its peers has failed if:

- It receives a *I have failed* health message from the peer.
- It does not receive any messages from the peer within the value set in **flipper!monitor_timeout**.

The Traffic Manager concludes that a peer has recovered when it starts receiving *I am healthy* messages from the peer.

Failover

When each Traffic Manager in a cluster determines that one of its peers has failed, the Traffic Manager may take over some or all of the traffic shares that the failed system was responsible for. The traffic distribution method determines how this is done.

Traffic IP Address Transfer (Single-Hosted Mode)

Each Traffic Manager in a cluster uses its knowledge of which machines are active to determine which Traffic IP addresses it should be running. The cluster uses a fully deterministic algorithm to distribute IP addresses across the machines:

- Because the algorithm is deterministic, the Traffic Managers do not need to negotiate between themselves when one of their peers fails or recovers.
- The algorithm is optimized to spread the distribution of Traffic IP addresses across the active Traffic Managers in a cluster, and to minimize the number of IP address transfers if a Traffic Manager fails or recovers.

When a Traffic Manager raises a Traffic IP address, it sends several ARP messages to inform adjacent network devices that the MAC address corresponding to the IP address may have changed. The Traffic Manager will send up to 10 ARP messages (tunable using the **flipper!arp_count** setting); the frequency of these messages is controlled by the **flipper!monitor_interval** setting (by default, the messages are sent at 0.5-second intervals).

Note that if a Traffic Manager detects that its own network connectivity has failed, it will immediately drop its Traffic IP addresses and broadcast *I have failed* health messages to its peers. This is in anticipation of other Traffic Managers in the cluster raising the interfaces when they realize that the first Traffic Manager has failed.

Traffic IP Address Transfer (Multi-Hosted Mode)

Each Traffic Manager in the Traffic IP Group deterministically chooses whether or not it should handle each packet, based on the source IP of that packet (and optionally the source port; see "Traffic Distribution" on page 107 for more details).

If a Traffic Manager fails, its share of the load is spread evenly between the remaining Traffic Managers. When it recovers, it takes equal shares of the load from its peers, thus ensuring that the traffic is always evenly distributed across the working machines in the Traffic IP Group.

Note: Multi-hosted IP functionality is not included with the Traffic Manager software by default. You can download and install it as an additional kernel module, and is supported on Linux kernels, version 2.6.18 and later. See the Traffic Manager documentation on the Brocade Web site (http://www.brocade.com) for more information on supported versions.

Traffic IP Address Transfer (RHI Mode)

If a Traffic Manager's fault tolerance checks fail, it lowers the addresses used in RHI traffic IP groups and withdraws route advertisements from the network.

If the network detects an inability to reach the designated active Traffic Manager in an RHI traffic IP group, routing decisions for the traffic IP address(es) use instead the next best available route using the lowest metric, such as to the designated passive Traffic Manager, or to a Traffic Manager hosting the same traffic IP address in another datacenter.

Recovering from Failure

When a failed Traffic Manager recovers, its share of traffic is transferred back to it.

Each time traffic shares are transferred from one Traffic Manager to another, any connections currently in that share are dropped. This is inevitable when a transfer occurs because a Traffic Manager fails, but may not be desirable when a Traffic Manager recovers.

In this case, you can disable the **flipper!autofailback** setting on the **System > Fault Tolerance** page of the Admin UI. When this is disabled, a Traffic Manager does not take any traffic when it recovers. Instead, the user interface displays a message indicating that the Traffic Manager has recovered and can take back its IP addresses.

When you want to reactivate the Traffic Manager, go to the **Diagnose** page and select the **Reactivate this Traffic Manager** link.

Alternatively, you can edit each of your traffic IP groups and set the recovered Traffic Manager to passive. Once you set it to passive in all of the groups, it will not need to take any shares of traffic; it will then reactive automatically, clearing the error state. In addition, no traffic will be lost because not traffic shares will have been transferred.

Debugging and Monitoring Fault Tolerance Activity

All state changes and IP address transfers are logged in the event logs of each relevant Traffic Manager. If email alerting is correctly configured (see CHAPTER 21), the Traffic Manager also sends an email message describing the state change and any IP address transfers that resulted to the system administrator.

For detailed debugging, enable the **flipper!verbose** setting. This causes each Traffic Manager to log every single connectivity test, broadcast message sent and broadcast message received, and is useful when determining why the fault tolerance behavior is not as expected.

See also the **Cluster Diagnosis** page in the **Diagnose** section of the Admin UI. This page informs you if any broadcast messages are not being received correctly, and provides a summary of the system status if an error has occurred.

For detailed information on your Route Health Injection status, see the **Diagnosis > Routing** page of the Admin UI.

Configuring BGP Connectivity

You can configure your Traffic IP Groups to use RHI as the IP distribution mode, using BGP as the routing protocol. To enable RHI for your Traffic IP addresses, use the **Services > Traffic IP Groups** page.

The settings in **System > Fault Tolerance > BGP Route Health Injection** control BGP connectivity between Traffic Managers in your cluster and the neighboring routing infrastructure:

- **bgp!enabled**: Whether BGP Route Health Injection is enabled.
- **bgp!as_number**: The Autonomous System (AS) in which your Traffic Manager cluster operates. Use a decimal value to represent the AS. This number applies to all Traffic Managers in your cluster (or each Configuration Location within a *Multi-Site Manager* cluster; see "Multi-Site Cluster Management" on page 393).

Configuring BGP Router IDs

By default, each Traffic Manager in the cluster joins the BGP routing domain using the address and subnet by which it is connected to its IPv4 default gateway. To override this behavior, you can set an alternate ID for each cluster member individually using **flipper!bgp_router_id**.

Managing BGP Neighbors

Unlike with OSPFv2, the Traffic Manager is unable to automatically discover its BGP neighbors. To enable your cluster to establish BGP sessions, you must first define the neighboring routers and then create applicable mappings between each cluster member and the desired neighbor.

To manage the BGP neighbors you can use in this mapping, click **Manage BGP Neighbors**. Use this page to define a new neighboring BGP enabled router, and to modify the parameters for existing neighbor definitions.

For each new router, input the following BGP connection parameters:

Field	Description
BGP Neighbor name	An identifying name
IP Address	The IPv4 address of the neighbor
AS Number	The AS number for this neighbor. This might be the same AS as your proposed corresponding Traffic Manager (in which case the session uses iBGP), or it might be a different AS altogether (in which case the session uses eBGP).
Keepalive	The interval, in seconds, between health message transmission.
Holdtime	The minimum interval, in seconds, that must elapse after the last received keepalive before this neighbor assumes the mapped Traffic Manager is no longer active.
Advertisement Interval	The interval, in seconds, that must elapse between advertisements made to this neighbor.

Field	Description
Authentication Password	(optional) The per-BGP session password that your mapped Traffic Manager must use with this neighbor. **Note**: Linux kernel versions earlier than 2.6.20, or any kernel built without the relevant option (CONFIG_TCP_MD5SIG) enabled, might not support this setting. For further clarification, see your system administrator or support provider.

After you have defined your BGP neighbors, click back to the **System > Fault Tolerance** page to create the mappings to your individual Traffic Managers.

Important: Before you can use BGP, you must configure the neighboring BGP routers to establish sessions with the selected Traffic Managers. This effectively corresponds with the neighbor definitions you have just created at the Traffic Manager end of the connection.

Click **Update** to apply your mappings and configuration. The settings on this page are replicated across all members of the Traffic Manager cluster.

The Traffic Manager reports all router peering and connection failures in the event log. The applet in the Admin UI and the Diagnose page both reflect the current status.

Configuring OSPFv2 Connectivity

You can configure your Traffic IP Groups to use RHI as the IP distribution mode, using OSPFv2 as the routing protocol. To enable RHI for your Traffic IP addresses, use the **Services > Traffic IP Groups** page.

The settings in **System > Fault Tolerance > OSPF Route Health Injection** control OSPFv2 connectivity in your Traffic Manager cluster:

- **ospfv2!enabled**: Whether OSPFv2 Route Health Injection is enabled.
- **ospfv2!area**: The OSPFv2 area in which your Traffic Manager cluster operates. You can use a decimal or IPv4 address format.
- **ospfv2!area_type**: The type of OSPFv2 area in which your Traffic Manager cluster operates. OSPFv2 requires all routers in a specific area to use the same type.

- **ospfv2!hello_interval**: The interval at which the Traffic Manager sends OSPFv2 "hello" packets to the network.
- **ospfv2!router_dead_interval**: The number of seconds that must elapse before the Traffic Manager declares a silent router down.

Note: If you disable OSPFv2, or shut down the Traffic Manager, it first de-advertises Traffic IP addresses by sending out a MaxAge Link State Advertisement (LSA) to all adjacent routers.

The Traffic Manager supports the use of two separate OSPFv2 authentication key/secret pairs ("a" and "b"). This mechanism enables you to maintain authentication across your network infrastructure in the event of your needing to replace one of the key/secret pairs. When updating the authentication key in use, you can keep the old key active until the new key has been provisioned across the network.

The Traffic Manager supports OSPFv2 authentication types 0 (none) and 2 (MD5). MD5 is used when at least one active key is configured.

- **ospfv2!authentication_key_id_a**: The OSPFv2 authentication key ID. Set to 0 to disable the key.
- **ospfv2!authentication_shared_secret_a**: The OSPFv2 authentication shared secret (MD5). Set to blank to disable the key.
- **ospfv2!authentication_key_id_b**: The OSPFv2 authentication key ID. Set to 0 to disable the key.
- **ospfv2!authentication_ shared_secret_b**: The OSPFv2 authentication shared secret (MD5). Set to blank to disable the key.

Note: If both key pairs are set to 0 (disabled), the Traffic Manager does not use OSPFv2 authentication.

The Traffic Manager does not act as a general purpose OSPFv2 router.

Configuring OSPFv2 IP Addresses

By default, each Traffic Manager in the cluster joins the OSPFv2 routing domain using the address and subnet by which it is connected to its IPv4 default gateway. To override this behavior, the Admin UI provides the means to set an alternate IPv4 address for each cluster member individually using **flipper!ospfv2_ip**.

The individual IP addresses you specify are used by each Traffic Manager as the next hop addresses for OSPFv2 routes that are installed on neighboring routers for traffic IPs that the Traffic Manager is advertising.

Set **flipper!ospfv2_ip** to "0.0.0.0" to disable the Traffic Manager's routing software.

Configuring Neighborhood Monitoring

The Traffic Manager polls the routing network every 10 seconds to collect neighborhood data. Among this data is a list of routers the Traffic Manager is peered with. For OSPF, this is known as the *neighbor list*.

To perform neighborhood monitoring, you configure each Traffic Manager in your cluster with the list of neighboring router IP addresses from which it should expect to receive traffic. This list is compared to the discovered neighbor list in order to determine the health of the routing network.

To configure the expected neighbors, type a space or comma separated list of IPv4 addresses in **flipper!ospfv2_neighbor_addrs**. Use the default value %gateway% to mean the default IPv4 gateway that each Traffic Manager is connected to.

To disable neighborhood monitoring for a specific Traffic Manager, set **flipper!ospfv2_neighbor_addrs** to an empty list.

The Traffic Manager reports a warning in the event log if some of the expected routers are not peered, and reports an error if none of the expected routers are peered. The applet in the Admin UI and the Diagnose page both reflect the status at the last poll.

Note: Due to the latency involved in establishing peering, you might experience transient warnings or errors if you modify your OSPF configuration.

CHAPTER 7 Key Features in the Traffic Manager Administration Interface

The Home Page

The Home Page of the Traffic Manager Administration Interface provides a dynamically updating overview of your key configuration, traffic and status:

- The **Status Applet** displays a real-time indication of the health status of the cluster, and a real-time traffic chart and summary. If a problem of any kind is detected, the status light in the applet will change from green to either orange (a minor problem) or red (a major problem). Click the status light to go directly to the Diagnosis page.
- The **Login Information Banner** provides a record of the previous successful login using these credentials. Any previous failed attempts are also shown here.
- The **Traffic Managers** section displays the status of the traffic management devices in your cluster. If any of the devices develops a fault, it will be highlighted and an error message displayed.
- The **Services** section displays a list of the services you are managing; the protocol, port, virtual server name and a list of all pools used by the service[2]. The **Stop/Start** button can be used to stop or start a service. If any problems are detected with a virtual server or pool, these will be highlighted.
- The **Event Log** section displays the most recent event log messages from the Traffic Manager systems in your cluster. Click the **Examine Logs** button to display the full event log. For more information, see "The Event and Audit Logs" on page 372.

The home page updates automatically to show new event log messages and changes in Traffic Manager, virtual server, or pool status.

Note: The Traffic Manager presents an adapted home page layout in multi-cluster management mode, including an additional section for your configured *Locations*. For full details, see CHAPTER 28, "Multi-Site Cluster Management".

[2] Note: only pools that are explicitly referenced in the configuration, as a default pool, or named in a TrafficScript or RuleBuilder rule, are listed. If you select pools dynamically, using `pool.use($variable)` for example, these pools will not be displayed in this list

Services > Configuration Summary

With a number or virtual servers, pools and rules set up, your configuration can soon become complex. The **Config Summary** provides a single-page synopsis of the items you have configured and how they connect together.

To view the Config Summary, click the **Services** button and then the **Config Summary** tab. Your services are displayed in a table, initially grouped by port and virtual server, with each of the items used by the service displayed alongside it.

You can rearrange the table to focus in on a particular column or configuration object, and see clearly what configuration objects depend on it. For example, clicking on the **Pool** column shows you each pool name, and beside it the nodes it contains, and all the virtual servers and rules that use it.

This allows you to track exactly where each configured item is used in a complex setup.

Catalogs

The Catalogs are central repositories of objects, and classes of objects, that you can use for managing traffic. You can store the following object types:

Note: Some catalog types might be license dependent. See the Brocade Web site (http://www.brocade.com) for more information.

- **Locations catalog:** Contains location-based configuration objects used for Global Load Balancing (GLB) and the Traffic Manager's multi-site cluster management capability.
- **DNS Server**: Contains zonefiles and zone configurations used with the Traffic Manager's authoritative Domain Name Server (DNS) capability.
- **GLB Services catalog**: Contains GLB service configurations used by the Traffic Manager to load balance clients across different geographic locations.
- **Aptimizer catalogs**: Contains application scopes and profiles used by the Traffic Manager's Aptimizer web content optimization technology.
- **Rules catalog:** Contains TrafficScript and RuleBuilder rules.
- **Java Extensions catalog:** Contains Java Extensions and any supporting Java class files that the extensions require. These Java Extensions can be invoked from a TrafficScript rule.

- **Monitors catalog:** Stores health monitors you can use to check the correct operation of nodes in a pool.
- **SSL catalog:** Contains SSL resources: server and client certificates, certificate authorities and certificate revocation lists.
- **Authenticators catalog:** Contains definitions of remote LDAP authentication services. These services can be accessed from TrafficScript to look up information about a user and to verify their password.
- **Kerberos catalog**: Contains configuration resources needed by the Traffic Manager to allow it to participate in Kerberos realms.
- **Service Protection classes:** holds classes that define the policies and security measures used by the Traffic Manager to filter unwanted traffic.
- **Session Persistence classes:** Contains classes that manage session persistence information for client connections.
- **Bandwidth classes:** Contains bandwidth allocations that you can use to manage bandwidth usage.
- **Service Level Monitoring classes:** Contains classes that monitor node response time and conformance to agreed levels of service.
- **Rate Shaping classes:** Contains classes you can use to queue and rate-shape requests to impose maximum request rates.
- **Cloud Credentials classes:** Holds the information required to allow the Traffic Manager to communicate with cloud provider APIs.
- **Extra files:** Contains additional files and resources accessible from TrafficScript rules (using `resource.get()`), or used to provide error messages in pool configuration.

Virtual servers, pools and rules can each reference objects in the Catalog. If you edit an item in the catalog, the changes you make propagate to every service that uses the item.

To access the catalogs, click **Catalogs** in the top navigation bar. To edit or add to the items in a catalog, click the appropriate **Edit** link.

Each catalog type is described in more detail later in this document.

License Management

Traffic Manager documentation and product interfaces often make reference to the use of license keys to enable or disable product features and capabilities. Brocade (or

your designated support provider) generates these keys on a case-by-case basis, and such keys are normally tied to your specific product instance or service agreement.

For clusters of multiple Traffic Manager instances, you should have a suitable key (or set of keys) to cover the licensing requirement of the whole cluster. Unlicensed Traffic Managers operate in Developer mode with a suitable warning banner.

Important: Although having differently licensed Traffic Managers within a single cluster is possible, Brocade does not recommend this due to the potential for automatic cluster synchronization failure when unlicensed features on one or more cluster members have updates attempted upon them.

To view your currently installed license keys, see **System > Licenses**. This page displays each license key, identifiable by its serial number, and the Traffic Managers it is valid for. To view full details for a specific license, such as product variant, expiry date, and licensed feature set, click the fold-down arrow.

When multiple licenses are installed on a Traffic Manager instance, the Traffic Manager selects the license to use based on the following ordered criteria:

- Any installed license is preferred over Developer mode.
- Any authorized license is preferred over non-authorized licenses.
- Remotely authorized licenses (including Flexible License Architecture (FLA) type licenses) are preferred over non-remote licenses (perpetual licenses).
- A license with a richer set of features is preferred over a license with a lesser set.

Adding and Removing License Keys

The first time you access the Admin UI of a freshly installed and configured Traffic Manager, you will be presented with an **Unlicensed** page that provides the opportunity to either upload a license key or continue in Developer mode. Appliance variants of the Traffic Manager also provide the opportunity to upload licenses through the Initial Configuration wizard. For more details, see the version of the "Installation and Getting Started Guide" most applicable to your product variant.

Beyond this, the **System > Licenses** page is the location to add and remove individual keys. New licenses can be uploaded using the *Install new License Key* section at the foot of the page. Click the **Choose File** option to provide the location of a standard text file containing the license data, and click **Install Key**. If this operation is successful, your new license will appear in the list.

Unneeded or invalid keys present in the cluster can be removed by selecting the checkbox to the right of the offending license and clicking the **Remove Selected Keys** button. If you attempt to remove a key that will result one or more Traffic Managers

becoming unlicensed, the Traffic Manager will warn you of this and require you to confirm the action by overriding the warning. If you choose to proceed, the affected Traffic Managers will revert back to Developer mode.

System > Global Settings

The Traffic Manager software provides an extensive set of configurable global settings. You can access these by clicking the **System** button and then the **Global Settings** tab. They cover aspects of the Traffic Manager that are not related to individual services or traffic, and are divided into the following categories:

Category	Description
System Settings	Define the network and system settings for each Traffic Manager.
EC2 Account Settings	Control how the Traffic Manager interacts with the Amazon EC2 API.
Cache Settings	Control the behavior of the shared caches in each Traffic Manager.
FIPS 140-2 Configuration	Determines whether "FIPS Mode" is enabled. This mode allows the Traffic Manager to process traffic using a FIPS 140-2 cryptographic module.
SSL Configuration	Control the behavior of the SSL support.
SSL Hardware Support	Apply to any additional SSL hardware used by the Traffic Manager.
Logging	Control how the Traffic Manager performs logging for system events and virtual server access.
State Synchronization Settings	Control how cluster members share their state data.
Idle Connection Settings	Control how idle HTTP connections to nodes are managed.

Category	Description
Java Extension Settings	Control how the Java Extension process is initialized and managed.
Login and Security Settings	Control user login behavior (see "Login Security and Behavior" on page 370).
Aptimizer	Control global properties of the Traffic Manager's Aptimizer capability.
Other Settings	Control other miscellaneous Traffic Manager settings.

Clicking on the arrow beside each category unfolds the options for that category. When you have finished making changes, click the **Update** button. You can use the **Restore Defaults** button to restore all the global settings to the system defaults.

Important: Take care when modifying any of these settings. The default values have been chosen to give good performance on a wide range of systems, and to preserve the stability and security of the software. Configuring these settings incorrectly could compromise the stability and security of your system, and may impact performance.

System > Backups

You can use the Traffic Manager's Backup Management capability to make backups of your configuration, compare backups with each other and with the current configuration, and restore configuration from a previous backup.

To manage your configuration backups, click **System > Backups**.

You can also use the `zconf` command-line utility to perform a more fine-grained backup/import/export of individual configuration objects. See CHAPTER 27 for more details.

Brocade recommends you make a backup of your configuration before making a large configuration change, so that you can:

- Compare configurations to find out exactly what you have changed.
- Restore the earlier backup if you wish to abandon the changes you have made.

This feature also enables you to copy configuration backups from one Traffic Manager machine to another if, for example, you want to replace an older virtual machine or cloud instance with a newer version.

Important: Configuration backup files are specific to the Traffic Manager on which they are created, and as such are not part of the configuration automatically replicated between Traffic Managers in a cluster. If you terminate or delete a particular cluster member, any configuration backups stored on it are lost.

Making a Backup

Use the '**Create a Backup**' form to make a backup of your configuration. This backup is stored on the local Traffic Manager machine.

If the Web Application Firewall is licensed and enabled in your Traffic Manager deployment, you can optionally include the Application Firewall configuration in your backup by ticking the **App Firewall** box.

Restoring a Backup

Click a backup name from the table on the Backup Management page**.** Use the '**Restore Configuration**' option to replace the current configuration with the contents of the configuration backup.

A configuration backup will contain machine-specific information from the Traffic Manager it was taken from, such as Traffic IP Groups. Therefore, at this point you must decide whether or not to replace the current Traffic Manager's local machine configuration. You have two options:

- Use the machine-specific configuration from the backup

  If you are restoring a backup made from *the same* Traffic Manager, you will simply overwrite the local configuration with the backed-up configuration. Should you have made this backup on a *different* Traffic Manager, you will be presented with a mapping control. Here you can decide on the mapping for the machine configuration stored in the backup:

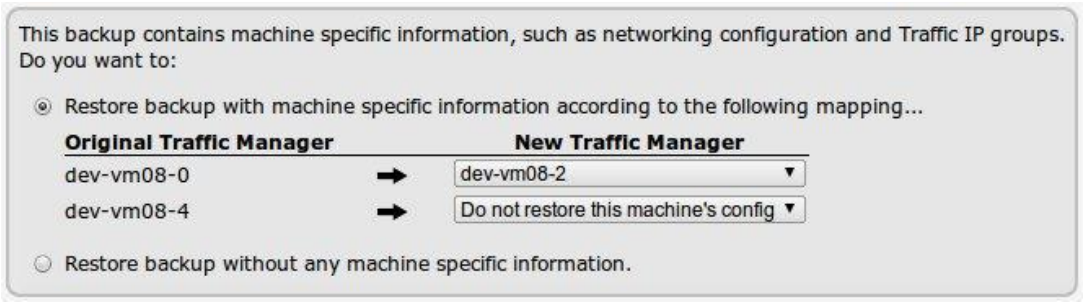

This backup contains machine specific information, such as networking configuration and Traffic IP groups.
Do you want to:
Restore backup with machine specific information according to the following mapping...
Original Traffic Manager
New Traffic Manager
dev-vm08-0
dev-vm08-2
dev-vm08-4
Do not restore this machine's config
Restore backup without any machine specific information.

Fig 19. An example cluster of two Traffic Managers

In this example, the backup contains configuration for a cluster of two Traffic Managers. Using this tool, you can decide which Traffic Manager is used for the restored configuration and which is ignored. Note that you can only create a one-to-one mapping between machine configurations.

- Restore the backup without any machine specific information.

  The machine-specific configuration stored in the backup is ignored, and the current Traffic Manager local machine configuration retained.

If the Web Application Firewall is licensed and enabled in your Traffic Manager deployment, you can optionally choose to also restore the Application Firewall configuration from your backup, if one exists, by clicking **Include Application Firewall configuration**.

Exporting a Backup

Configuration backups are stored on the local Traffic Manager machine. You can export a configuration backup for safekeeping.

Click a backup name from the table on the Backup Management page. Use the '**Export Configuration**' option to download the configuration backup to your local machine.

Important: Store this backup securely. It contains sensitive information, including SSL certificates.

Importing a Backup

You can import a previously exported configuration backup. Use the **Import a Backup** section to upload a configuration backup from your local machine.

When you import a backup, it is added to the list of configuration backups on the Traffic Manager machine. It does not replace the current configuration. You can then restore the backup you have just uploaded to replace the current configuration if desired.

System > Backups > Partial Backups

This section provides the same backup/restore functionality of a full backup, yet allows you to tailor the contents of the backup to contain only a subset of a full configuration. It can be accessed from the link on the **System > Backups** page.

This feature provides similar functionality to the `zconf` command-line utility described in CHAPTER 27, and can be useful when trying to copy specific services to other Traffic Manager clusters. Unlike a regular import, the configuration being imported is merged with the existing one instead of replacing it.

The page is separated into two main sections, one to handle the import and merge process, and another to enable exporting of partial backups.

Importing

This section of the page allows you to import a partial or full backup. To facilitate a partial import from the uploaded backup file, you can provide a suitable filter in the **Include Only** text box provided. The system will import any objects that match this filter, along with any objects upon which the selected objects depend. For example, the pools used by a selected virtual server will also be imported. Please refer to the **Filter formats** section below for more details about the format used.

After uploading a backup file, and optionally specifying a filter, you will be presented with a *Diff* (a list of differences) showing the changes that will be made to the system. You should review the Diff and satisfy yourself that everything is correct. Click **Apply Partial Backup** to commit the changes.

Important: Importing partial backups can lead to an inconsistent configuration if not handled correctly. Always review the *Diffs* presented when importing a partial backup. As with all backup and restore operations, it is strongly recommended that you create a full backup point before applying a partial backup.

Exporting

This section is used to download partial or full configuration backups. To facilitate a partial backup, you can set a suitable filter in the **Include Only** text box provided. The system will then export only the configuration objects that match this filter, along with their dependencies. Please refer to the **Filter formats** section below for more details about the format used.

Filter Formats

The **Include Only** filter is a space separated list of entries according to one of the following formats:

`[CONFIGURATION TYPE]` For example: `vservers`

or

`[CONFIGURATION TYPE]/[NAME]` For example: `vservers/Intranet`

or

`[CONFIGURATION FILE]` For example: `users`

Note: When a configuration name has one or more spaces in it, you should precede each space character with a forward slash (\). This will ensure that the space is treated as part of the name and not a separator. For example, `Intranet Master Service` would be entered as `Intranet\ Master\ Service`.

The filter allows the use of an asterisk (*) as a wildcard to represent zero or more characters. For example, typing "net*" matches "net", "net.cfg", and "network".

Note: You can include the special configuration file `settings.cfg` to back up your global settings (**System > Global Settings**), and the configuration file `users` to back up your local user settings (**System > Users**).

For example, to import all virtual servers, the monitor "Primary Database Monitor" and your global settings, enter the following filter:

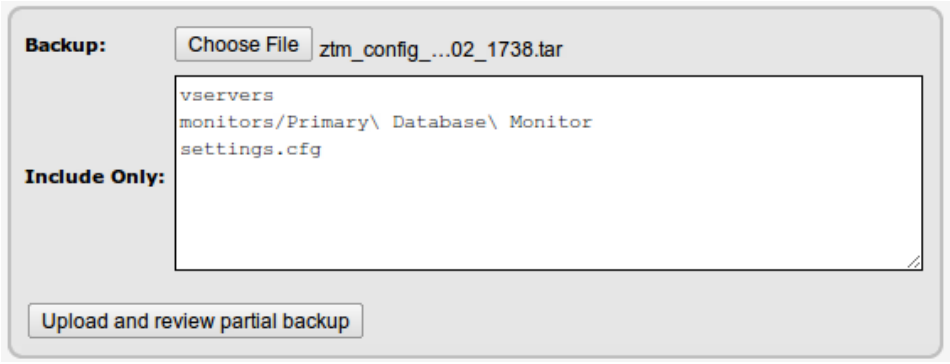


Fig 20. The Include Only filter for a partial backup import

The following table lists the configuration identifiers you can choose to include:

Identifier	Description
actionprogs	Action Programs
actions	Alerting Actions
activitymonitor	Current Activity Graphing Data Settings
appliance	Appliance Management
aptimizer/profiles	Aptimizer Profiles
aptimizer/scopes	Aptimizer Scopes
auth	Admin Interface / Admin Server Authenticators
authenticators	TrafficScript Remote Authenticators

Identifier	Description
ApplicationFirewallConfig	Application Firewall Configuration
auth	Authenticators
bandwidth	Bandwidth Classes
cloudcredentials	Cloud Credentials
custom	Custom Configuration Sets
dnsserver/zonefiles	DNS Server Zone Files
dnsserver/zones	DNS Server Zones
events	Alerting Event Types
extra	Miscellaneous Files
flipper	Traffic IP Groups
groups	User Groups
jars	TrafficScript Java Extensions
licensekeys	License Keys
locations	Multi Site Management and Global Load Balancing Geographic Locations
monitors	Monitors
persistence	Session Persistence Classes
pools	Pools
protection	Service Protection Classes
rate	Rate Shaping Classes
rules	TrafficScript Rules
scripts	Monitor Scripts
services	Global Load Balancing (GLB) Services

Identifier	Description
servlets	Java Extension Servlets
slm	Service Level Monitoring Classes
ssl/cas	SSL Certificate Authorities
ssl/client_keys	SSL Client Certificates/Keys
ssl/dnssec_keys	DNSSEC Keys
ssl/server_keys	SSL Server Certificates/Keys
supplementarykeys	Supplementary License Keys
vservers	Virtual Servers
zxtms	Traffic Managers

Including the Web Application Firewall in a Partial Backup

To include the Web Application Firewall (WAF) in a partial backup, you must also include the Global Settings file. In addition, WAF must be enabled in your Traffic Manager configuration before you create the backup.

In other words, with WAF enabled, add the following to the "Include Only" filter in the **Export** section:

- `settings.cfg`
- `ApplicationFirewallConfig`

Equally, to import WAF configuration through the Partial Backups page, ensure you add both files to the "Include Only" filter in the **Import** section.

Important: Restoring the `settings.cfg` file to your active Traffic Manager configuration in this way overrides all Global Settings configuration, not just WAF-specific key values. Brocade recommends creating a backup of this file before restoring a copy of it through a partial backup.

Activity Monitoring

Several visual traffic monitoring tools allow you to keep track of the current and past activity on your system. To access these tools, click the **Activity** button on the top bar of the Admin Server interface.

Activity > Current Activity

The **Current Activity** page shows you a real-time graph of the traffic activity on your system. You can customize the data extensively.

You can choose some standard sets of data to plot, traffic values such as current bandwidth, number of connections the Traffic Manager processes or hardware values such as free memory or spare CPU time. Much of this information can be split by virtual server, by pool, or by node. You can also dictate the size of the graph, the timescale, and choose between linear and logarithmic axes, or a pie chart. You can graph across all the machines or one at a time.

When you have chosen your settings, click **Plot Data** to view the graph. You can also download the data as a `.tsv` (tab-separated variable) file for your own analysis.

To further customize the data on the graph, click the **Change Data** button. This takes you to the **Current Activity > Edit** page where you can specify the data to plot.

In the **Global Values** section you can choose statistics to monitor and plot for the system as a whole. The **Virtual Server Values**, **Pool Values** and **Node Values** sections allow you to plot data split into individual instances of these. You can view data transfer and connection counts as well as DNS and SNMP request data, content compression and HTTP rewrites.

The **Service Protection Class Values** section allows you to plot information about requests rejected by service protection classes. You can view data for all or a subset of your service protection classes, and split the data by the reason for rejection, such as the connection count from a single IP address being too high or the request containing disallowed content.

When you have chosen the settings to plot, you can save them to use again later. When you have finished, click **Apply**.

Monitoring Performance Using SNMP

Simple Network Management Protocol (SNMP) is an open standard for network monitoring and management. It can allow you to monitor devices on a network and gather performance data.

The Traffic Manager supports multiple SNMP versions: **v1**,**v2c** and **v3**.

The Traffic Manager allows you to monitor performance using an SNMP tool such as HP OpenView™. You can extract information about the traffic flow for your own analysis.

Note: The Traffic Manager is also capable of sending asynchronous SNMP notifications, known as SNMP Traps. See CHAPTER 21 for more details.

To configure SNMP, click the **System** button and then the **SNMP** tab. Here you can download the relevant Traffic Manager *MIB* (Management Information Base) for your chosen SNMP version. The MIB is a database of the information used by SNMP, and comes in two SMI (Structure of Managed Information) versions:

- **SMIv1**: for SNMPv1 clients only
- **SMIv2**: for SNMPv2c and v3 clients only

Either click the link to view the MIB in your browser, or right-click to save the contents to a file on your computer.

Common SNMP Settings

To enable SNMP, set the `snmp!enabled` radio button to **Yes**.

You can configure common settings such as the port and bind IP address the SNMP service should listen on, and the clients that should be allowed to access the service.

The standard port for SNMP is 161, but if you wish to use an unprivileged port then 1161 is a good alternative.

SNMPv1 and SNMPv2c Settings

You can configure the following settings:

Setting	Description
snmp!community	In order to function correctly, the SNMP server must have a suitable community string. This is the equivalent of a password, and identifies authorized SNMP requests. It is not secure, as it is transmitted in the SNMP request as plain text, so you should not use a sensitive secret as the community string. The default value for many SNMP clients is "public". Since the Traffic Manager can restrict access to the SNMP service, the strength of this secret is not a major concern. Leaving the community string blank/empty will cause all SNMPv1 and SNMPv2c commands to be rejected.

SNMPv3 Settings

This section allows you to specify the authentication and privacy settings for accepting and responding to SNMPv3 commands.

Note: One ingredient involved in the generation of cryptographic keys from passwords is the so-called *Engine ID*. The Traffic Manager generates this value using a combination of the Zeus[3] enterprise OID (7146) and the first MAC address on the machine it is running on. The correct Engine ID value for the Traffic Manager you are connected to is displayed in this section.

All SNMPv3 communication is checked against the user settings that follow, and commands whose security parameters do not exactly match are rejected. (However, `snmpEngineID` discovery messages, which are never authenticated and which may be sent with an empty username, are permitted and responded to).

You can configure the following settings:

Setting	Description
snmp!username	This is the SNMPv3 user's username, required for all SNMPv3 commands and communication. If you set this to blank/empty, all SNMPv3 commands will be rejected. The Traffic Manager requires and permits a single SNMPv3 user account to be defined. Authentication and privacy settings for this user are defined according to the fields that follow, and are based on the "User-based Security Model" described in RFC3414 (http://www.ietf.org/rfc/rfc3414.txt).

[3] Zeus is the name of the product upon which the Traffic Manager was originally based.

Setting	Description
snmp!security_level	The SNMPv3 user's choice of whether to use authentication, and whether to use privacy. SNMPv3 *commands* must obey this choice, or they are rejected. SNMPv3 *responses* obey this choice. Selecting **Authentication** here means that commands and responses are cryptographically signed with a key derived from snmp!auth_password, and this signature is included in the message. Incorrectly signed commands or responses are detected and rejected. This is intended to prevent an imposter masquerading as a trusted party, as long as the password remains secret. Additionally selecting **Privacy** here means that parts of commands and responses are encrypted using DES with a key derived from the snmp!priv_password. This is intended to prevent an eavesdropper from reading the content of messages, as long as the password remains secret.
snmp!hash_alg	This is the SNMPv3 user's choice of cryptographic hash algorithm, required for SNMPv3 commands and used for responses. Supported hash algorithms are MD5 and SHA-1. See RFC 3414 for details (http://www.ietf.org/rfc/rfc3414.txt). If you set snmp!security_level to "No Authentication, No Privacy", this setting is not used.
snmp!auth_password	The SNMPv3 user's password for authentication. This is a shared secret; you should configure your SNMP client user account with the same authentication password. The authentication key for SNMPv3 communication is derived from this password. If you set snmp!security_level to "No Authentication, No Privacy", this setting is not used.

Setting	Description
snmp!priv_password	The SNMPv3 user's password for privacy (message encryption). This is a shared secret; you should configure your SNMP client user account with the same privacy password. The privacy key for SNMPv3 communication is derived from this password. Re-using your authentication password here is not recommended for security reasons. If you set snmp!security_level to "No Authentication, No Privacy", or "Authentication only", this password is not used.

Custom Performance and Event Monitoring

The `counter.increment()` TrafficScript function is used to increment one of 10 built-in counters.

You can retrieve the values of these counters using SNMP, and you can graph them in the Activity Monitor.

These counters can be used to record custom events that you detect using TrafficScript. For example:

- If you perform authentication in a TrafficScript rule, you could count the number of authentication failures.
- If a particular class of request is particularly resource heavy, you could chart the number of times that request is made.
- If you perform security checking, for example, for a known Web worm signature, you could count and chart the number of times the signature was found.

You can increase the number of built-in SNMP counters by changing the **snmp_user_counters** value located in **System>Global Settings>Other Settings**.

Activity > Historical Activity

The Traffic Manager records the system's activity for the past 90 days. By clicking on the **Historical Activity** tab, you can view this data in graphical form.

You can choose to display the hits per minute or the bytes processed per second, and split the results by virtual server, pool or node. Timescales to plot range from 1 hour

to 30 days, and you can choose to plot the data only for certain virtual servers, pools, or nodes. You can dictate the chart size and axis type, as for the Current Activity graph.

When you have chosen your settings, click **Plot Data** to view the graph. You can also download the data as a `.tsv` (tab-separated variable) file for your own analysis.

You can change the number of days that data is stored for by changing the **statd!days** setting located in **System>Global Settings>Other Settings**.

Activity > Map

The map provides a view of the geographic origin of requests being handled by your traffic management system.

The map is designed to be interactive and tools are provided to move around and zoom into a level of detail required for your infrastructure. You can center the map on a geographic location of your choice by holding down the left-hand mouse button and dragging the map to your required destination.

Each small dot that appears on the map represents a user who has issued a request for a particular service.

Activity > Connections

Note: This functionality described in this section is individually license-controlled. Some licenses provide a **basic** functionality mode only, with simple connection information, fewer customization options and a single filter. A **full** mode that supports all the features described here is available under the *advanced real-time analytics* license option. Contact your support provider if you wish to purchase this option.

Basic Mode

The **Connections** page gives you the ability to monitor current and recent connections being handled by the Traffic Manager, in date order (most recent first). This is a snapshot of connections from a point in time, where all connections after that time will not be displayed until the **Refresh Snapshot** button is pressed.

You can show and hide field columns using the **[+]** button in the top left of the connections table. This displays a drop-down list of fields that the table is currently displaying. Clicking on each field in this list toggles it between being displayed or hidden.

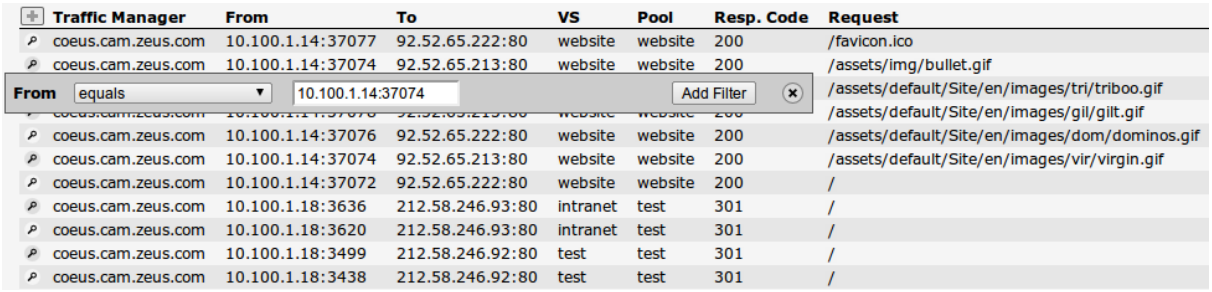
A simple filter can be applied to the displayed data to show connections only to a specific node (the 'To' column). This is a single-field version of the filter functionality available in the **full** mode. Refer to the usage instructions described below.

Full Mode

In this mode all of the above features are present, with additional connection detail and customization now available. You can apply a sort to the displayed data by clicking on the column heading you wish to sort by. A small arrow next to the sorted column indicates the direction of sort. You can toggle this between ascending and descending by clicking on it.

In addition to the node filter, you can now filter the connections by any field or data in the table. This can be achieved via the drop-down box in the filter section at the top of the page, or by clicking directly on one of the data values in the table. This second method produces an in-line *add filter* dialog that allows you to filter away all connections that do not share the value of the selected field.

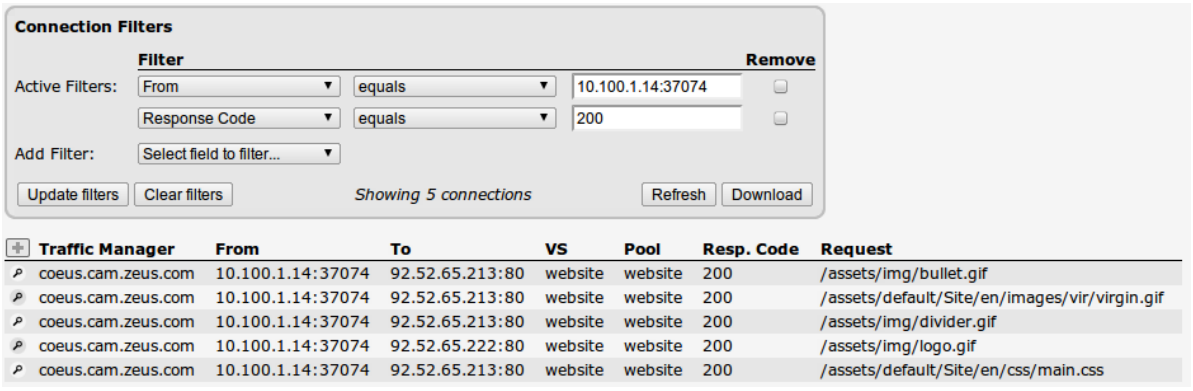
For example, suppose you want to see requests that originate from a specific IP address and port that result in an HTTP response code of 200. First click the IP address you wish to use in the **From** column. A filter popup will appear, populated with the selected IP:



Traffic Manager	From	To	VS	Pool	Resp. Code	Request
coeus.cam.zeus.com	10.100.1.14:37077	92.52.65.222:80	website	website	200	/favicon.ico
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/img/bullet.gif
coeus.cam.zeus.com	10.100.1.14:37076	92.52.65.222:80	website	website	200	/assets/default/Site/en/images/dom/dominos.gif
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/default/Site/en/images/vir/virgin.gif
coeus.cam.zeus.com	10.100.1.14:37072	92.52.65.222:80	website	website	200	/
coeus.cam.zeus.com	10.100.1.18:3636	212.58.246.93:80	intranet	test	301	/
coeus.cam.zeus.com	10.100.1.18:3620	212.58.246.93:80	intranet	test	301	/
coeus.cam.zeus.com	10.100.1.18:3499	212.58.246.92:80	test	test	301	/
coeus.cam.zeus.com	10.100.1.18:3438	212.58.246.92:80	test	test	301	/

Fig 21. Adding a filter to the "From" column

Click the **Filter** button to apply the described logic. This filter is added to the *Active Filters* display at the top of the page. Now click the value "200" in the **Resp. Code** column and apply a filter to this. You should now see only the desired connections.



Traffic Manager	From	To	VS	Pool	Resp. Code	Request
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/img/bullet.gif
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/default/Site/en/images/vir/virgin.gif
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/img/divider.gif
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.222:80	website	website	200	/assets/img/logo.gif
coeus.cam.zeus.com	10.100.1.14:37074	92.52.65.213:80	website	website	200	/assets/default/Site/en/css/main.css

Fig 22. Active connection filters

Adjustments can be made to the active filters by making a change and then clicking the **Update Filters** button to redisplay the connection data. To remove a filter, tick

the checkbox next to the criteria that you want to remove, and click **Update Filters**. All filters can be removed by clicking **Clear Filters**.

The **Refresh** button can be used to update the results snapshot, and the **Download** button provides a **.tsv** (tab separated value) file of the results for analysis in other applications.

Furthermore, clicking on the magnifying glass icon to the left of each line provides additional details for a specific connection:

Section	Description
Request tracing	This section gives you a timeline of internal connection processing events. This can show you how long the entire request took, how much of that time was spent processing TrafficScript rules, and other useful information. This information will recorded only if the **request_tracing!enabled** key is enabled for the virtual server that processed the connection. See "Connection Analytics" on page 69 for further information.
Request details	This section shows you all of the request headers that the request contained.
Response details	This section shows you all of the response headers (including those that may have been added by TrafficScript or verbose cache logging).

Note: You can increase the size of the Connections snapshot using the **System > Global Settings** page; look in the **Logging** section for the **recent_conns** key.

Activity > Draining Nodes

If you wish to remove a node from the system, for instance for maintenance or upgrade, you can set it to ***drain***. The node will continue to handle any existing sessions, but the Traffic Manager will not send it any more connections. The **Draining Nodes** section shows information about any nodes you are currently draining.

For each draining node, the page shows the time since the last connection, and the current number of active connections. You can update the information by clicking **Reload This Page**.

When all the active sessions have expired, it is safe to remove the node; you can do this via the **Pools > Edit** page or use the **Remove a Node** wizard.

For further information, please refer to the description of node draining in the *Draining and Disabling Nodes* section of CHAPTER 5.

Activity > View Logs

Virtual servers may be configured to log all transactions to a Request Log. In the **Virtual Server > Request Logging** page (see the *Request Logging* section of CHAPTER 4), you may configure what information is logged, and where the log files are written to.

The **Activity > View Logs** page allows you to watch the request logs in real-time as records are appended to them.

Note: Traffic Manager virtual and cloud instances are self-contained and have a dedicated, self-managed log partition. Log files are stored in this partition, archived and deleted automatically. You do not need to configure the location where request log files are stored.

Note: Log files can be viewed using the **Activity > View Logs** page, and can be downloaded using the **Activity > Download Logs** page. You can also use the Traffic Manager Control API to download and manage log files on Traffic Manager virtual/cloud instances.

Cloud Credentials

When using Traffic Manager software in a cloud environment, the Traffic Manager might require authentication credentials in order to make API calls to the cloud provider. Primarily, pools set to use autoscaling will require API credentials to enable the scaling mechanism within the cloud. This section allows you to record such credentials in a catalog object, based on the requirements of the chosen cloud provider API. For more details about pool autoscaling, see "Autoscaling" on page 93.

The Traffic Manager provides a number of built-in cloud API's upon which to base your credential set:

- VMware vSphere
- Amazon EC2

- Rackspace Cloud

Additional custom cloud APIs can be added by uploading appropriate executable scripts through the **Catalogs > Extra Files** facility. Please refer to the Traffic Manager section of the Brocade Community Web site (http://community.brocade.com) for further details.

To create a new set of cloud credentials:

1. Click **Catalogs > Cloud Credentials**
2. Enter a name in the **Name** field
3. Select the **Cloud API** you wish to use.
4. The remaining fields differ depending on the API chosen. They typically include an ID, password/passcode, and additional authentication information.
5. Click the **Create Cloud Credentials** button.

The table below provides details for each of the fields presented when you create a new set of credentials:

Setting	Description
Name	The name used to identify this set of credentials within the Traffic Manager's configuration.
Cloud API	The selected cloud provider API.
ID / Name (cred1)	The username or ID of the cloud provider account to be used for API calls.
Auth Key / Password (cred2)	The password associated with the username/ID.
Token / Server (cred3)	Some cloud providers also require an authentication token or additional item in order to make API calls. VMware vSphere requires a vCenter hostname/IP to accept API calls. This extra information can be specified here.

Setting	Description
update_interval	The Traffic Manager periodically queries the cloud provider for the status of all instances running on behalf of the user. This is necessary in order to be up-to-date when nodes are added or removed by external systems. The `update_interval` determines how often (in seconds) these status calls are made. Note that some clouds impose a limit on the number of such calls that can be made per minute.
change_process_timeout	The maximum amount of time (in seconds) a change process can take. For example, when a request is made to the cloud API for node creation/destruction, this setting specifies how long to wait for the request to complete.

IAM Roles in Amazon EC2 Credentials

Amazon EC2 deployments can use Identity and Access Management (IAM) roles in place of locally stored credentials. If you are creating an EC2 credentials set, choose whether to specify an Access Key ID and Secret Key or leave these fields blank to instead use an IAM role.

Note: The Traffic Manager uses any credentials specified here in preference to any assigned IAM role, regardless of whether the role exists.

For more information on IAM roles, including how to create and manage roles, see the AWS documentation at: http://aws.amazon.com/documentation/.

The Web Application Firewall

The Brocade Virtual Web Application Firewall is an enterprise-level Web Application Firewall that provides attack detection and protection for the latest generation of mission-critical Web applications.

It enables centralized security monitoring, reporting and alerting and provides custom protection for your Web applications and infrastructure against external attacks.

Note: This is an optional capability of the Traffic Manager that is only activated through the appropriate license key upgrade. Contact your support provider for more details.

Overview

The Application Firewall is an optional component of the Traffic Manager. Once licensed and enabled in your cluster, it becomes fully integrated into the overall capability of the Traffic Manager software.

The Traffic Manager provides the infrastructure and overall control of your Web services, passing traffic to the Application Firewall when instructed to do so.

This chapter provides an overview of the Application Firewall and how it interacts with the Traffic Manager. For full instructions on configuring and using the Application Firewall, see the *Brocade Virtual Web Application Firewall User Guide* and on-line help.

Enabling the Application Firewall

Important: Do not enable the Application Firewall component on a mixed cluster of software and non-software instances of the Traffic Manager. This configuration is not supported.

To use the Application Firewall, obtain a suitable license key from your support provider. Once you have installed the new license, the Traffic Manager activates the **System > Application Firewall** page in the Admin UI. Use this page to enable, disable, and configure the Application Firewall component.

As with other configuration changes, enabling or disabling the Application Firewall on one Traffic Manager in your cluster automatically causes all other cluster members to make equivalent changes to their configuration.

Important: Enabling the Application Firewall imposes an automatic restart of the Traffic Manager software. Brocade recommends that you undertake this procedure at a time where the impact on your services is minimal.

Application Firewall Features in the Traffic Manager Admin UI

To control and configure the Application Firewall, use the button provided in the Traffic Manager navigation bar. This button operates as a toggle between the Traffic Manager and Application Firewall user interfaces.

Note: You must enable the Application Firewall component to make this button visible.

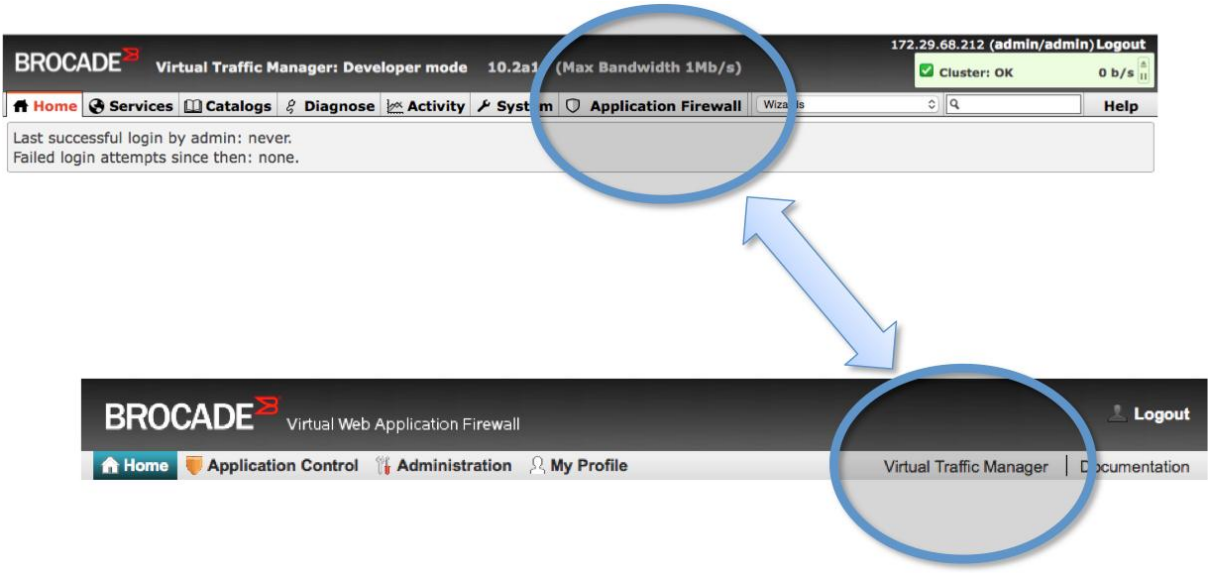


Fig 23. Links between the main Traffic Manager and Application Firewall navigation bars

There are additionally a number of features in the main Traffic Manager Admin UI that govern Application Firewall communication and behavior:

- The **System > Application Firewall** page contains sections to enable and disable the Application Firewall component. It also contains low-level Application Firewall settings, including what network ports it uses and the number of processes it runs.
- A "Restart Application Firewall..." button is added to the Software Restart section of the **System > Traffic Managers** page. Certain configuration changes might require a restart of the Application Firewall without requiring the Traffic Manager itself to be restarted.
- The Basic Settings section of the **Services > Virtual Servers > Edit** page contains a control to enable or disable the Application Firewall for each of your HTTP services.
- The Rules catalog contains a new built-in TrafficScript rule, "Application Firewall Enforcer", that the Traffic Manager uses to maintain communication with the Application Firewall service.
- The Traffic Manager provides status messages in its event and audit logs relating to the Application Firewall service, particularly pertaining to its current running state. Any software stops, starts and restarts are recorded, along with specific configuration key changes and monitor errors. The Application Firewall also maintains its own local event log for attack and other event information relating to the security of your services. You can view this log through the Application Firewall Admin UI. See the *Brocade Virtual Web Application Firewall User Guide* for further information.

- An Application Firewall configuration section is present on the **Diagnose** page, which provides more detailed descriptions of any firewall problems that occur within your cluster. Should communication between the Traffic Manager and the Application Firewall be interrupted for any reason, the **Status Applet** shows that an error has occurred. Click on this error indicator to access the Diagnose page where you can see which particular problem has occurred and determine the appropriate action to remedy the situation.

The System > Application Firewall Page

This page contains configuration options for Application Firewall performance and communication.

Note: During normal operation, you do not need to modify any of the settings in this section. The descriptions given here are for information only.

The Application Firewall requires a dedicated base port for communication between the Decider process and the Enforcer TrafficScript rule.

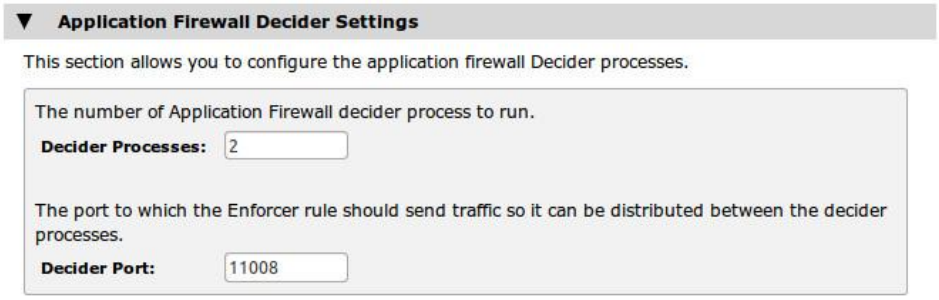

Application Firewall Decider Settings
This section allows you to configure the application firewall Decider processes.
The number of Application Firewall decider process to run.
Decider Processes:
2
The port to which the Enforcer rule should send traffic so it can be distributed between the decider processes.
Decider Port:
11008

Fig 24. The Decider settings

The factory default port is suitable for the majority of deployments, but you can specify a different value if this port is unavailable on your system. The Application Firewall restarts automatically if you change the settings on this page.

Use the "Decider Processes" setting to tune the performance of the Application Firewall according to the number of CPU cores available on your system. To handle more traffic and thus improve the performance of your system, increase this value to instantiate additional Decider processes. However, running more processes than there are free resources can inhibit performance and degrade the service. In these circumstances, reduce the number of processes to help increase overall system performance.

Brocade recommends that you set the number of decider processes to be the same as the number of CPU cores available on the server with the fewest CPU cores in your Traffic Manager cluster. For example, if you have two Traffic Managers in your

cluster, one with four CPU cores and one with two CPU cores, set the number of decider processes to 2.

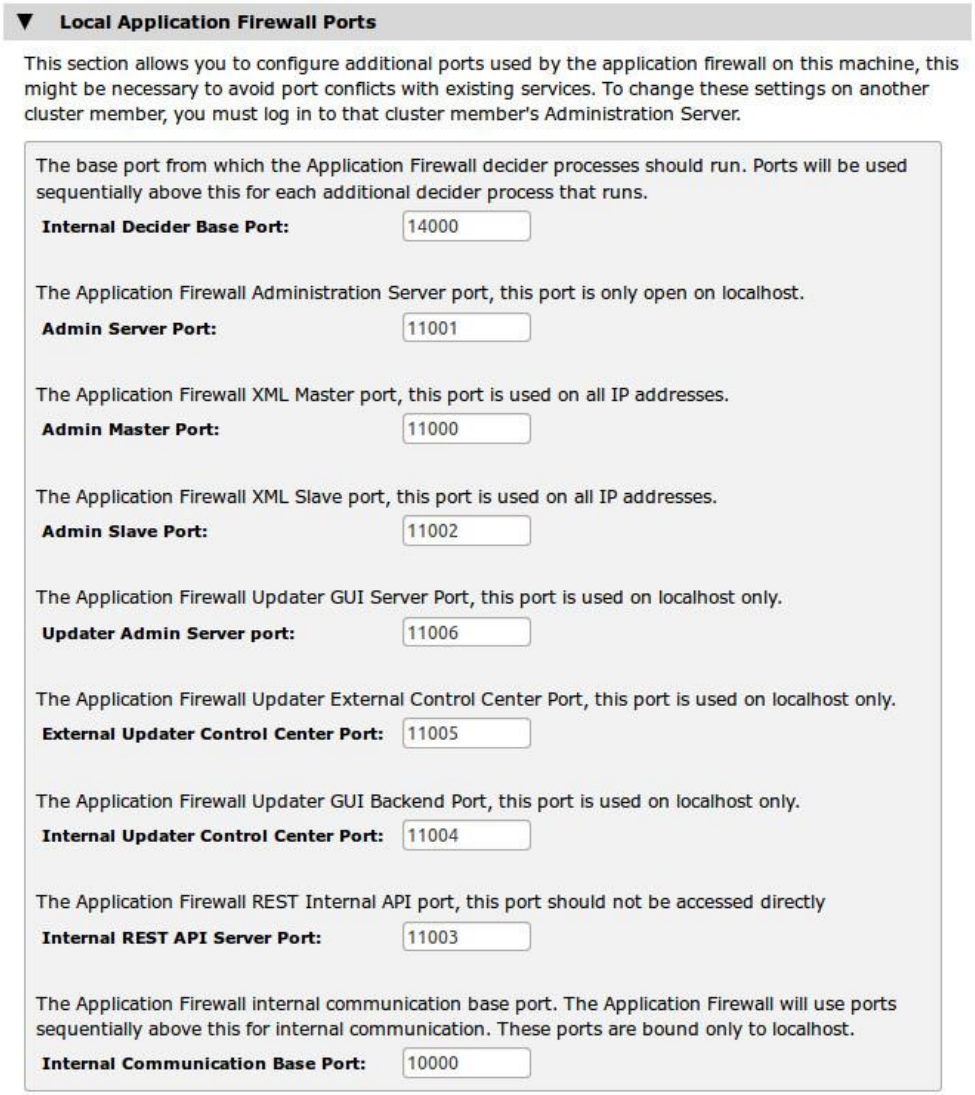

Local Application Firewall Ports
This section allows you to configure additional ports used by the application firewall on this machine, this might be necessary to avoid port conflicts with existing services. To change these settings on another cluster member, you must log in to that cluster member's Administration Server.
The base port from which the Application Firewall decider processes should run. Ports will be used sequentially above this for each additional decider process that runs.
Internal Decider Base Port: 14000
The Application Firewall Administration Server port, this port is only open on localhost.
Admin Server Port: 11001
The Application Firewall XML Master port, this port is used on all IP addresses.
Admin Master Port: 11000
The Application Firewall XML Slave port, this port is used on all IP addresses.
Admin Slave Port: 11002
The Application Firewall Updater GUI Server Port, this port is used on localhost only.
Updater Admin Server port: 11006
The Application Firewall Updater External Control Center Port, this port is used on localhost only.
External Updater Control Center Port: 11005
The Application Firewall Updater GUI Backend Port, this port is used on localhost only.
Internal Updater Control Center Port: 11004
The Application Firewall REST Internal API port, this port should not be accessed directly
Internal REST API Server Port: 11003
The Application Firewall internal communication base port. The Application Firewall will use ports sequentially above this for internal communication. These ports are bound only to localhost.
Internal Communication Base Port: 10000

Fig 25. Additional communications ports

The Application Firewall requires a number of other ports for internal communication purposes. Under normal circumstances, you do not need to modify these default values. However, to resolve port number clashes, change the required port settings to alternative values on this page.

The Enforcer and Decider

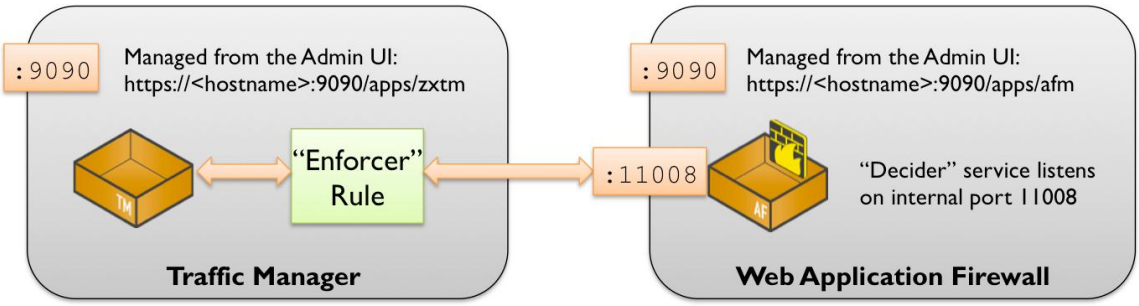


Fig 26. The Traffic Manager's Enforcer TrafficScript rule forwards requests to the default local port 11008 for inspection by the Decider service. The Decider returns a "Permit" or "Deny" decision and the Enforcer rule applies the decision.

The *Enforcer* is a built-in TrafficScript rule (entitled "Application Firewall Enforcer") that enables communication between the Traffic Manager and the Application Firewall. It captures all HTTP requests and responses and forwards them to the *Decider* service running on the Application Firewall for further consideration.

The Decider uses a set of rules stored in a configuration database to evaluate HTTP requests and to make decisions on the actions to be carried out. The Enforcer then implements these decisions whereby each request or response is accepted, modified or denied as appropriate.

See "The Enforcer Rule" on page 148 for more information on the Enforcer rule.

The Enforcer Rule

The Traffic Manager communicates with the Application Firewall via the TrafficScript rule system, applying certain logic to the requests and responses generated by traffic to and from your Web applications. When you enable the Application Firewall, a special rule entitled "Application Firewall Enforcer" (the Enforcer) is created to instruct the Traffic Manager to forward HTTP traffic to the Decider service. To attach this rule to your virtual servers, enable the **Application Firewall** setting on the **Virtual Server > Edit** page.

You cannot enable, disable, remove, or edit the Enforcer rule in the same way as other rules, but you can move it to alter the TrafficScript rule execution order. This is useful if you need to process requests or responses first. For example, you might have an additional rule that whitelists certain requests ahead of executing the Enforcer, as per the following TrafficScript:

```
connection.data.set( "enforcer.whitelist", 1 );
```

Like other rules, the Enforcer is in the *Rules Catalog*. As mentioned earlier, you cannot delete it, but you can use the *Save As* functionality to make a copy of it.

Note: Such copies look like, and provide the same options as, the original Enforcer rule. However, you should use them like traditional rules. In other words, you assign them to virtual servers, pools and rules in the normal way. Only the original Enforcer rule is assigned using a virtual server's *Application Firewall* setting.

You can also set a number of configurable data items for the Enforcer. Click the rule name, or the *Edit* link, to access its settings page. The actual TrafficScript text is not immediately visible - click the **Rule text** link to display it. Note that it is in a binary format and is only partially human-readable.

The top of the rule text shows a number of configurable parameters. The Traffic Manager automatically updates the rule text with any modifications you make:

- **allowonerror**: Set to 1 to allow all traffic if the Decider is un-contactable. (default: 0)
- **backend_timeout**: The number of seconds to wait for the Decider before timing-out. (default: 5)
- **bypass_enforcer_max**: Set to 0 to block requests larger than **enforcer_max_body_size** from bypassing the Application Firewall. (default: 1)
- **bypass_file_types**: A space separated list of file extensions to exempt from the Application Firewall. (default: "css js png jpg gif")
- **debug:** Set to 1 to enable debugging mode. Use this option to provide a more verbose error output to assist in tracking down problems. (default: 0)
- **discard_on_reject:** Set to 1 to close rejected connections without sending an error message.
- **enforcer_max_body_size:** The maximum body size, in bytes, that the Application Firewall processes. Requests larger than this limit are blocked or bypassed depending on the setting of **bypass_enforcer_max**. (default: 2097152)

Note: You should not normally need to modify these settings. Doing so may inhibit the ability of the rule to function correctly, so only proceed if you are fully aware of the consequences.

User Management

The Traffic Manager maintains user and group access with the Application Firewall. It creates a new Application Firewall user with the same level of authorization as a

member of the **admin** group in the Traffic Manager Admin UI[4]. Performing usual administrative tasks does not require additional access privileges or login processing. Click **Logout** in either UI to log the user out of both applications.

Important: If you delete a user from the Traffic Manager, it is not automatically removed from the Application Firewall. An administrator must perform this step manually.

Members of other user groups in the Traffic Manager, such as **Demo** or **Monitoring**, only gain access to the various Application Firewall UI features if the correct permission settings exist on the **System > Users > Groups > Edit Group** page. The Traffic Manager provides settings in the **System** section for this purpose:

- **Application Firewall:** Controls access to the **System > Application Firewall** page in the Traffic Manager Admin UI.
- **Application Firewall > Administration:** Controls access to the Application Firewall Admin UI (through the toolbar button shown earlier).

Note: You can override the automatic admin user creation mechanism by pre-defining user accounts in the Application Firewall Admin UI (if, for example, you want to specify limited access, or use a non-admin group). A user with the same name on the Traffic Manager will then have only the desired access in the Application Firewall Admin UI.

Updating Your Software

The Application Firewall includes a capability known as the *Updater* that is responsible for handling updates to the Application Firewall software component. You can access the Updater through the **Administration > Cluster Management** tab of the Application Firewall Admin UI. See the *Brocade Virtual Web Application Firewall User Guide* and on-line help for more information.

[4] Specifically, an Application Firewall user record is not created UNTIL you first access the Application Firewall Admin UI from the Traffic Manager Admin UI. If the username already exists, no new user is created and the existing user record is used instead.

CHAPTER 8 TrafficScript Rules

This chapter describes the system of TrafficScript rules used on the Traffic Manager. It explains how to create and apply new rules and provides some examples.

Note: Some variants of the Traffic Manager do not support TrafficScript rules or XML capabilities, or have limited support (RuleBuilder only). Full support can be obtained via a software or license key upgrade.

The Traffic Manager section of the Brocade Community Web site contains a large amount of examples and documentation describing how to use TrafficScript rules to solve a range of network and application problems. It is located at http://community.brocade.com.

Overview

You can customize the Traffic Manager using your own traffic management rules. These rules are created using a scripting language called ***TrafficScript***.

The Traffic Manager executes TrafficScript rules whenever it receives a new connection or network request, whenever it receives a response from a node, and at the logical completion of a transaction. The rules can inspect the incoming and outgoing data in the connection, and other aspects such as the remote client address.

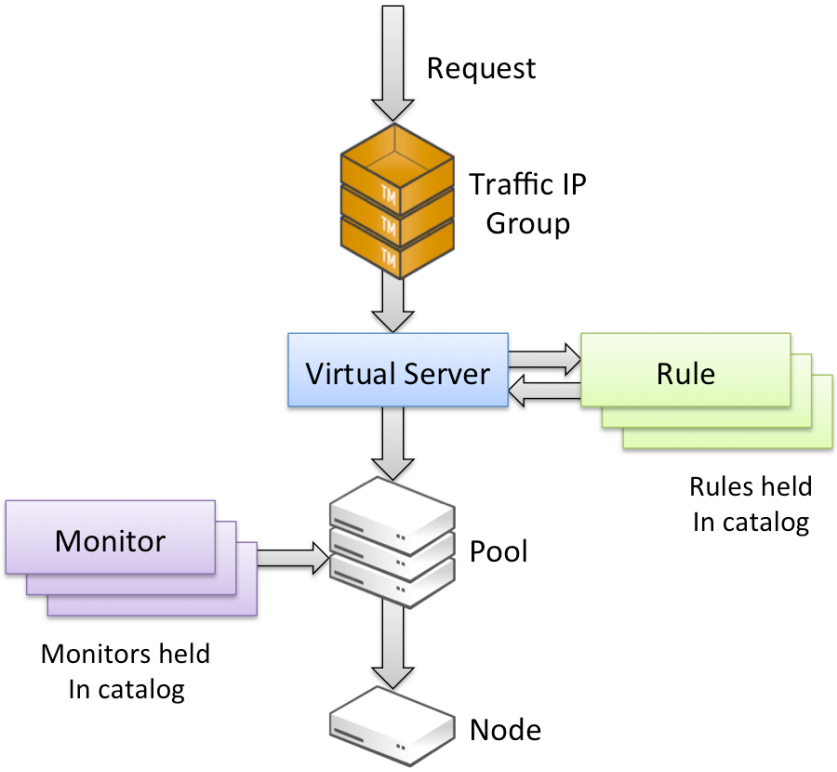


Fig 27. Analyzing, modifying and monitoring traffic using TrafficScript rules

A TrafficScript rule can modify the request or response (for example, rewriting the URL or headers in an HTTP request), set session persistence parameters, selectively log transactions, or inform the Traffic Manager how to route the request to the most appropriate pool.

This makes it possible to control precisely how the Traffic Manager manages your traffic by using rules designed to meet your own hosting requirements.

TrafficScript Example

The following TrafficScript rule can be used with HTTP requests. It handles the request as follows:

- Requests for `www.example.co.uk` are rewritten to `www.example.com`.
- Requests for `.jsp` pages are routed to a set of application servers (the pool named `JSPServers`).
- Requests for URLs beginning "`/secure`" are only allowed during office hours.

```
# Rewrite host header if necessary
if( http.getHostHeader() == "www.example.co.uk" ) {
  http.setHeader( "Host", "www.example.com" );
}

$path = http.getPath();

# Give .jsp requests to the "JSPServers" pool
if( string.endsWith( $path, ".jsp" ) ) {
  pool.use( "JSPServers" );
}

# Deny access to /secure outside office hours
if( string.startsWith( $path, "/secure" ) ) {
  if( sys.time.hour() < 9 || sys.time.hour() >= 18 ){
    connection.discard();
  }
}
```

The next rule can be used with HTTP responses. It processes the response as follows:

- If the status code is 404 or 5xx, retry the request a maximum of three times.

- If the response contains references to `www.example.co.uk`, rewrite it by changing these references to `www.example.com`.

```
$code = http.getResponseCode();
if( $code == 404 || ( $code >= 500 && $code != 503 ) ) {
   # Not retrying 503s here, because they get retried
   # automatically before response rules are run
   if( request.getRetries() < 3 ) {
      # Avoid the current node when we retry,
      # if possible
      request.avoidNode( connection.getNode() );
      request.retry();
   }
}

# We're only going to process text/html responses, so
# break out of the rule if the response is of a
# different type...
if( http.getResponseHeader( "Content-Type" ) !=
             "text/html" ) break;

# getResponseBody automatically de-chunks and
uncompresses
# the response if required
$response = http.getResponseBody();

if( string.contains( $response, "www.example.co.uk" ) )
{
  $response = string.regexsub( $response,
             "www.example.co.uk", "www.example.com", "g"
);
  http.setResponseBody( $response );
}
```

TrafficScript Documentation

The syntax and functions of the TrafficScript language are documented fully in *Brocade Virtual Traffic Manager: TrafficScript Guide*. You can access this guide from the Traffic Manager product page on the Brocade Web site at:

http://www.brocade.com/en/products-services/application-delivery-controllers/virtual-traffic-manager.html

The TrafficScript edit page also includes a quick reference link to the functions available. To view the reference, click the **TrafficScript Reference** link when editing a rule.

For a description of how to create rules, see "Creating a Rule in the Catalog" on page 155.

Applications of Rules on the Traffic Manager

Virtual servers typically use rules to select appropriate pools to handle requests. The rule can inspect any part of the request, possibly modify it, and decide which pool should handle the request.

You can use a rule to dictate session persistence information to a pool. After inspecting the request, the rule can use the `connection.setPersistenceKey()` function to provide a string to persist on. This string is used by the Universal session persistence method to identify the session the request belongs to.

Rules can be used to check the response from the server and modify it, or even retry the request (if possible) if a transient error was detected.

Rules can override the classes assigned to a connection by the virtual server or the pool. This way, they can specify custom behavior for each connection; connections to a slow resource can be given a longer response time tolerance for example. Classes you can assign in this way include *Service Level Monitoring*, *Session Persistence* and *Bandwidth Management*.

Service protection classes can use rules. If you have associated a service protection class with your virtual server, it inspects the incoming packets. The class may use a rule to check the packet for a match with known worms or viruses. This rule is executed before the main processing of the virtual server is carried out.

Session persistence and service protection applications of rules are covered in chapters CHAPTER 12 and CHAPTER 15 respectively.

Using a Rule on the Traffic Manager

TrafficScript rules are stored in the Rules Catalog. You can use the Rules Catalog to create rules, upload them from an external source, modify or duplicate them, and delete unused rules as required.

To access the Rules Catalog, click **Catalogs > Rules**.

You can configure a virtual server to execute one or more rules from the catalog each time it receives a new request or response, or at completion of a transaction. This

way, several different virtual servers can use the same rule, and modifications to the rule take effect on all virtual servers.

To use a rule, first create or upload a new rule in the catalog. Then configure your virtual server to use the rule (click **Services > Virtual Servers > Rules** and add the rule using the settings provided).

Creating a Rule in the Catalog

To create a new rule:

1. Click **Catalogs > Rules Catalog**.
2. In the "Create new rule" section, enter a name for your rule.
3. To write the rule in TrafficScript, select **Use TrafficScript Language**. To use the RuleBuilder, click **Use RuleBuilder**.
4. Click **Create Rule**.

Using the RuleBuilder

Note: Your license key might restrict the availability of certain RuleBuilder options and features and the ability to use TrafficScript. If you encounter problems using some of the features described here, consult your Support Provider who can advise you on the best course of action to upgrade your software or license.

The RuleBuilder allows you to select conditions on the request to be examined, and actions that follow if any or all of these conditions are met.

You can add conditions based on the client's IP address and port, HTTP parameters, cookies, or the URL requested. Whenever you create a condition, extra text boxes appear for you to fill in the parameters for that item.

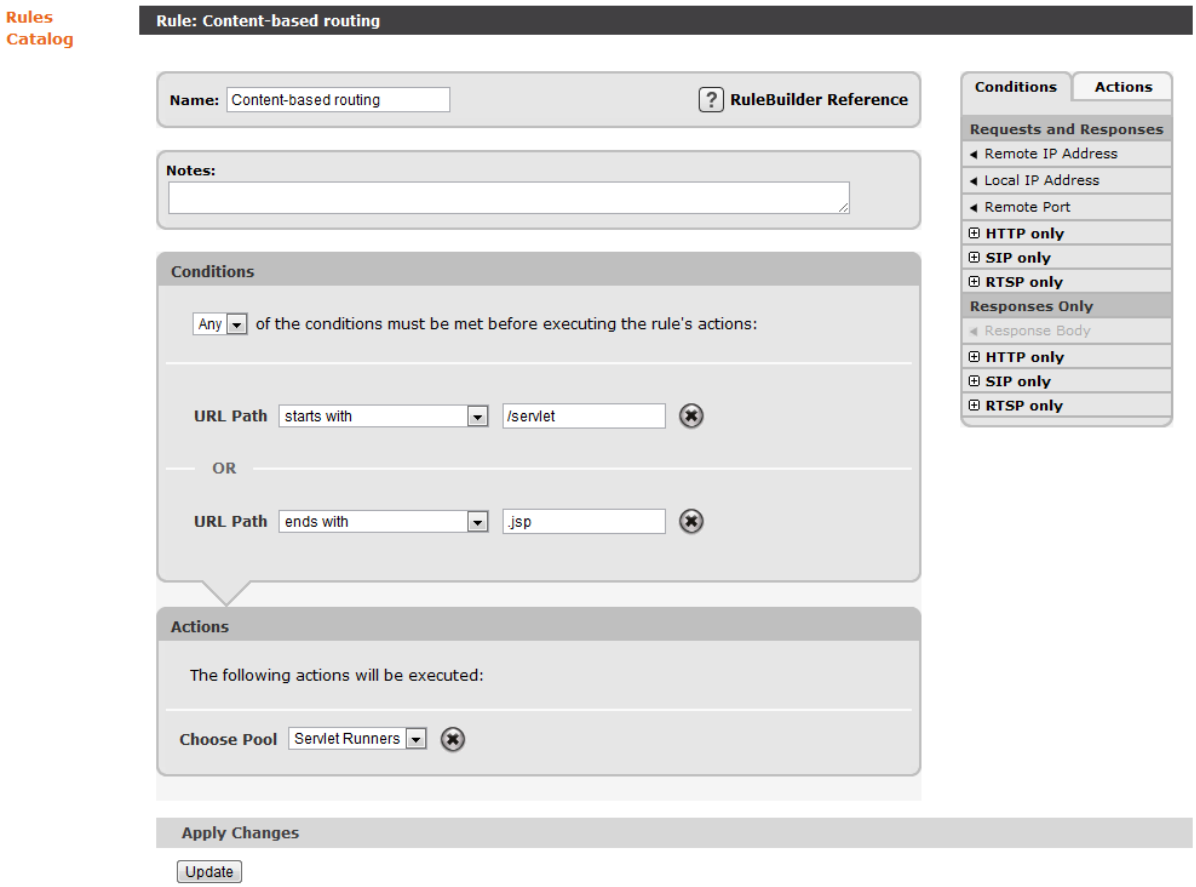


Fig 28. RuleBuilder main screen

You can then select a series of actions for the rule to carry out. Parts of the request can be altered and the Traffic Manager can write log messages before a final action is performed. The possible final actions are to choose a specified pool, or to drop the connection; not more than one final action can be carried out, but a rule is not required to have a final action.

For example, you might construct a rule such as:

```
IF
        URL Path  starts with    /servlet
OR
        URL Path  ends with      .jsp
THEN
        Choose Pool: Servlet Runners
```

Requests whose URL path does not start with `/servlet` or end in `.jsp` (such as requests for static HTML content) are ignored by this rule. A virtual server applying this rule can use a separate rule to deal with requests for other pages, or assign them to its default pool.

When you are happy with your rule, click **Update**.

You can also convert your rule to TrafficScript. The TrafficScript language gives you a much wider range and structure of possible conditions and actions. You can click **Preview Rule as TrafficScript** to see what the TrafficScript rule would look like, and use the **Convert Rule** button to convert your rule permanently.

When you begin to use the Traffic Manager's rules, you could start by using the RuleBuilder to implement simple rules. Examine the corresponding TrafficScript rule to familiarize yourself with the TrafficScript syntax. When you need to write a complex rule, you could use the RuleBuilder to prototype a simple version; once you have reached the limits of the capabilities of the RuleBuilder, proceed by converting the rule to TrafficScript for further editing.

Special variables

There are several special variables that can be included in the text boxes for RuleBuilder which are expanded to TrafficScript functions. These allow the generation of more dynamic rules, with extra flexibility.

For example, if you create a rule that sets the HTTP header 'X-Forwarded-For' to the value '%REMOTE_IP%', then the header will be set to the IP address of the client connecting to the Traffic Manager.

The special variables are as follows:

- %REMOTE_IP%– replaced with the remote IP of the client connection (TrafficScript function request.getRemoteIP());
- %REMOTE_PORT%– replaced with the remote IP of the client connection (TrafficScript function request.getRemotePort()).

Writing a TrafficScript Rule

The TrafficScript editing page shows a form where you can enter a text description of the rule, type the rule and check the syntax before compiling it. You may wish to write the rule in a separate text editor before pasting it in.

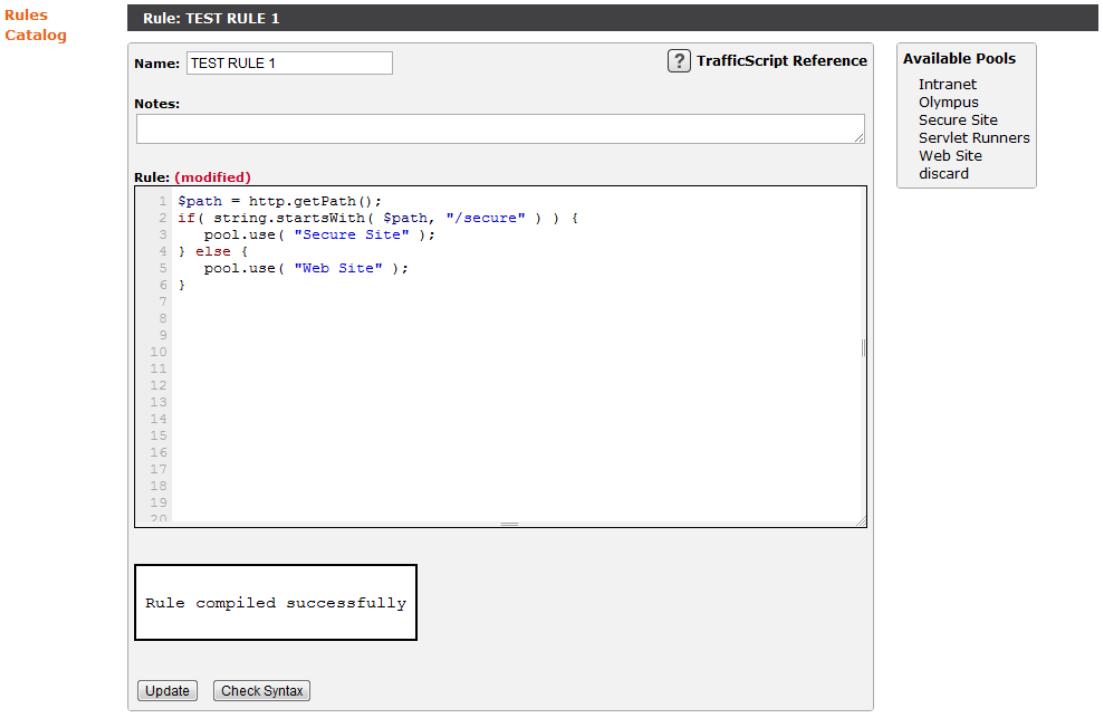


Fig 29. Editing a rule in the TrafficScript editor

Click the **TrafficScript Reference** link to view the quick function reference, and use the **Check Syntax** button to check your rule. When you have finished, click the **Update** button.

Note: Some TrafficScript functions and RuleBuilder conditions and actions are appropriate only in request rules, response rules, or transaction completion rules specifically. For example, a function that modifies a parameter of a request has no effect if used in a response rule (because the request has already been submitted to a node). Equally, any transaction completion rule that attempts to use a function with side-effects for the connection (for example, http.setResponseHeader(), connection.discard()) is disallowed. See the Online Help or **TrafficScript Guide** for descriptions of each TrafficScript function and RuleBuilder conditions and actions.

Uploading a Rule to the Catalog

The Traffic Manager supports uploading of previously created TrafficScript rules into the Catalog. Your rules should be in plain text, and the Traffic Manager uses the full filename (including extension) as the rule name.

Note: The Traffic Manager performs a syntax check on the uploaded rule file. Warnings and errors are displayed on the Rule Catalog page, the Diagnose page, and in the Event Log.

To upload a rule to the Catalog

1. Click **Catalogs > Rules Catalog**.
2. In the "Upload an existing rule" section, click **Choose file** to select a rule file from your local filesystem.
3. Choose whether to overwrite existing rules that have the same name as your filename.
4. Click **Upload Rule**.

Applying a Rule to a Virtual Server

A newly added rule is not yet in use by the Traffic Manager. To use a rule, add it to your virtual server configuration.

To configure a virtual server to use a rule

1. To access the **Virtual Servers > Edit** page for your virtual server, click the **Services** button and then the **Virtual Servers** tab. Click the name of your virtual server.
2. Click the **Rules** link. This presents you with lists of request rules, response rules, and transaction completion rules currently in use for your virtual server.
3. To add a new rule to a list, use the *Add rule:* option in each section. Choose a rule from the drop-down list and click **Add rule**.
4. Rules are executed in a specified order. If the first rule does not make a final decision about a transaction, the second rule is tested, and so on. Use the *drag handle* at the left side of each rule bar to move a rule within the list.

   For most protocols, you can specify whether the rule should be executed just once (against the first request or response), or against every request and response in the protocol dialogue.

   In RTSP and SIP the rules are automatically executed on each individual request or response that passes through the Traffic Manager.

   This option is not necessary for HTTP virtual servers because HTTP is a single request-response protocol, and requests within a *keepalive* connection are processed independently.

You can test the effect of a new rule by enabling and disabling it for your test virtual server.

Example Rules

Routing by Content Type

This example inspects the URL path in an HTTP request. It chooses a pool to handle the request based on the value of the URL path.

Set up a pool for each of these groups called `windows`, `sun` and `linux` respectively. The following rule directs network traffic according to the type of content.

```
$path = http.getPath();
if( string.endsWith( $path, ".jsp" )) {
  pool.use( "sun" );
} else if( string.endsWith( $path, ".asp" )) {
  pool.use( "windows" );
} else {
  pool.use( "linux" );
}
```

Restricting Access Based on Time of Day

This example only allows access to a particular service during office hours (between 9am and 6pm, Monday to Friday). It discards all connections that occur outside these times.

```
$dayofweek = sys.time.weekDay();
$hourofday = sys.time.hour();

# $dayofweek: Sunday is 1, Saturday is 7
# $hourofday: office hours are between 9am and 5:59pm
if( $dayofweek == 1 ||
    $dayofweek == 7 ||
    $hourofday < 9  ||
    $hourofday >= 18 ) {
  connection.discard();
}
```

In practice, it may be more appropriate to direct restricted traffic to a separate “error pool” of servers rather than just dropping the connection without warning. The servers in the error pool would be configured to return an appropriate error message before closing the connection. The procedure for doing this depends on the protocol being balanced.

Customer Prioritization

This example inspects the cookie in an HTTP request. It uses the value of the cookie to determine which pool to select. One pool is faster than the other because it contains machines reserved for premium users.

A company has a customer base divided into “gold” and “silver” membership. It wishes to give priority to the “gold” customers and has five servers, `yellow`, `green`, `blue`, `black` and `purple`.

Two pools are created: `standard`, for the “silver” customers, containing machines `yellow`, `green` and `blue`; and `premium`, for the “gold” customers, which includes all five of the servers. Thus `black` and `purple` are only available to the “gold” customers.

The site uses a cookie login system, with the customer type encoded in the cookie. The Traffic Manager can differentiate between membership levels and send traffic to the correct pool:

```
$cookie = http.getHeader( "cookie" );
if( string.contains( $cookie, "gold" )) {
  pool.use( "premium" );
} else {
  pool.use( "standard" );
}
```

Managing Levels of Service

This example tags premium customers with a 'premium' service level monitoring class, and directs them to the 'premium' pool.

Non-premium customers can share the premium pool if the premium SLM class is functioning within its tolerance, but are directed to the 'standard' pool if the premium SLM class is running too slowly.

```
# $isPremium could be based on the presence of a
# login cookie, the contents of the shopping cart,
# or even the remote address of the client.

if( $isPremium ) {
  connection.setServiceLevelClass( "premium" );
  pool.use( "premium" );
} else {
```

```
  if( slm.conforming( "premium" ) == 100 ) {
    pool.use( "premium" );
  } else {
    pool.use( "standard" );
  }
}
```

Routing Based on XML Traffic

TrafficScript includes support for parsing XML documents using XPath, an industry-standard language used to query XML documents.

XML documents are used by SOAP-based protocols such as Web services, and enable complex data to be exchanged and understood automatically without user intervention.

An XML document is organized into a tree structure of *nodes*[5]. Each node may contain a piece of data, or other nodes. XPath can navigate through these nodes to extract specific data from the XML document; this data can then be used by the Traffic Manager to make routing decisions on the traffic.

The XPath 1.0 specification is available at http://www.w3.org/TR/xpath.

Example: Google Search Request

The Google™ search engine has a Web services interface that accepts SOAP requests for search queries. A request for a search for Brocade Communications would consist of an HTTP POST containing the following XML body data:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/"
  SOAP-
ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encod
ing/"
  xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/1999/XMLSchema">
```

[5] Note that these are unrelated to the Traffic Manager's back-end nodes.

```
   <SOAP-ENV:Body>
      <namesp1:doGoogleSearch
xmlns:namesp1="urn:GoogleSearch">
         <key xsi:type="xsd:string">googleUniqueID</key>
         <q xsi:type="xsd:string">Brocade Communications</q>
         <start xsi:type="xsd:int">0</start>
         <maxResults xsi:type="xsd:int">10</maxResults>
         <filter xsi:type="xsd:boolean">false</filter>
         <restrict xsi:type="xsd:string"/>
         <safeSearch
xsi:type="xsd:boolean">false</safeSearch>
         <lr xsi:type="xsd:string"/>
         <ie xsi:type="xsd:string">latin1</ie>
         <oe xsi:type="xsd:string">latin1</oe>
      </namesp1:doGoogleSearch>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Note that the SOAP body contains a "doGoogleSearch" node. This contains the parameters of the search request.

An Internet service may implement or proxy doGoogleSearch requests, and the Traffic Manager may be used to manage the traffic to this service.

For example, it may be necessary to split doGoogleSearch requests according to the specified maximum number of results. If maxResults is greater than 100, the request is to be sent to pool googleLarge; otherwise it should be sent to pool google.

A TrafficScript rule can use the functions xml.XPath.MatchNodeSet() and xml.XPath.MatchNodeCount() to query the SOAP request body and test XML nodes:

```
# Read the entire body of the SOAP/HTTP request
$body = http.getBody();

# XML parameters lie in the "urn:GoogleSearch" XML
# namespace:
$googlens = "xmlns:googlens=\"urn:GoogleSearch\"";

# Test for the presence of a "doGoogleSearch" node.
# If present, get the value of the "maxResults"
# parameter and choose the pool
```

```
if( xml.XPath.MatchNodeCount( $body, $googlens,
    "//googlens:doGoogleSearch" ) ) {

  $maxResults = xml.XPath.MatchNodeSet(
    $body, $googlens,
    "//googlens:doGoogleSearch/maxResults/text()" );

  if( $maxResults >= 100 ) {
    pool.use( "googleLarge" )
  } else {
    pool.use( "google" );
  }
}
```

CHAPTER 9 TrafficScript Authentication Support

Note: Some versions of the product do not support TrafficScript, so this functionality might not be available. Full support can be obtained via a software or license key upgrade.

Overview

The Traffic Manager provides support for remote user authentication against external services. Each time a user connects to a service provided through the Traffic Manager, their credentials can be validated against the records held on a remote server.

To achieve this, you must first create remote authentication service definitions in the Authenticators Catalog (**Catalogs > Authenticators**). Authenticators created in this manner can be accessed through the `auth.query()` function from within a TrafficScript rule. This rule can then be added to a virtual server handling the service to be authenticated.

Configuring Authenticators

The Traffic Manager currently provides support for LDAP authentication only.

Remote LDAP authenticator records can be created on the **Catalogs > Authenticators** page of the Admin UI. To create a new authenticator, provide a name, host and port in the boxes provided. Clicking **Create Authenticator** will create the record, and allow you to access the remaining LDAP configuration keys. These are shown below:

Basic Settings

Setting	Description

Setting	Description
Name	The identifying name given to this authenticator. This name will be used in the `auth.query()` function call, within an authenticating TrafficScript rule. If the authenticator is renamed here, any rules referencing this authenticator will be automatically updated to use the new name.
Host	The host-name or IP address of the LDAP server to connect to.
Port	The port of the LDAP server to connect to.
Note	A description of this authenticator.

LDAP Settings

Setting	Description
ldap!bind!dn	The Distinguished Name (DN) of the 'bind' user. The bind user is used to contact the LDAP server and search for the record belonging to the user being authenticated. The bind user must have permission to search for and read user records on the LDAP server. If no bind user is specified then the Traffic Manager will attempt an anonymous login to search for the user being authenticated.
ldap!bind!password	The password for the bind user specified by the ldap!bind!dn setting.

Setting	Description
ldap!filter	A filter used to extract the unique user record located under the base DN. The string `%u` will be replaced by the username supplied when the authenticator is invoked. Examples of common LDAP filters include `sAMAccountName=%u` for Active Directory, or `uid=%u` for some Unix LDAP schemas.
ldap!filter!basedn	The base Distinguished Name (DN) under which the Traffic Manager will search for the record of the user being authenticated. The entries for all users that are to be authenticated by this LDAP authenticator must appear under the DN specified here. A typical base DN might be `OU=users, DC=mycompany, DC=local`.
ldap!attr	If the Traffic Manager finds a record for the user being authenticated on the LDAP server it can fetch back additional information from that record. This information can be used to perform additional checks on the user being authenticated, such as restricting access based on which group the user belongs to. To fetch back specific attributes from the user's record, a space- or comma-separated list of attribute names can be specified in the ldap!attr setting. To fetch back all attributes from the user's record, set ldap!attr to '*'. If the setting is blank, no additional attributes will be retrieved from the server. Any attributes retrieved from the user's record will be available in the return value from the TrafficScript function that requested the authentication.

Setting	Description
ldap!ssl	This setting determines whether or not the connection to the LDAP server will be SSL-encrypted. The method by which the SSL connection is established can be specified in the ldap!ssl!type setting.
ldap!ssl!type	This setting specifies the method by which an SSL connection to the LDAP server is established. It is used only if ldap!ssl is set to **Yes**. The available methods are: ▪ **LDAPS**: The Traffic Manager will establish a secure connection to the LDAP server before any LDAP messages are sent. ▪ **Start TLS**: The Traffic Manager will use the LDAPv3 Transport Layer Security extension to establish a secure connection to the server.
ldap!ssl!cert	When connecting to the remote LDAP server over SSL, the Traffic Manager will ensure that the server's certificate is signed by the certificate authority specified by this setting. If the server sends a certificate that is not signed by the certificate authority specified here then an error will be returned to the TrafficScript function using the authenticator. If no certificate authority is specified then the server's certificate will not be validated.

Configuring the TrafficScript Rule

To use remote authentication within a virtual server, you should assign an authentication rule to it as a request rule. This rule should contain a call to the function `auth.query()` with the arguments shown here:

```
auth.query( authenticator, user, [password] );
```

This queries the named `authenticator` for information about `user` and, if supplied, checks that `password` matches the password on record for that user. It returns a hash containing two values, `OK` and `Error`, which are set according to the results of the verification. The result can also contain additional information returned by the authenticator, such as the Distinguished Name of the user that was queried. Please refer to the **TrafficScript Reference** for more details.

TrafficScript rules are created on the **Catalogs > Rules** page of the Admin UI. For more details, please refer to CHAPTER 8.

The example below shows how you might use the `auth.query()` function to provide user verification based on a previously created Authenticator called '`ldap`':

```
# Verify the user's password using an LDAP
# authenticator called 'ldap'
$auth = auth.query( "ldap", $user, $pass );
if( $auth['Error'] ) {
   log.error(
      "Error with authenticator 'ldap': " .
      $auth['Error']
   );
   connection.discard();
} else if( !$auth['OK'] ) {
   # Unauthorised
   http.sendResponse( "403 Permission Denied",
      "text/html", "Incorrect username or password",
      ""
   );
}

# Allow through members of the 'admin' group using
# the 'group' attribute returned by the authenticator
if( $auth['group'] != "admin" ) {
   http.sendResponse( "403 Permission Denied",
      "text/html",
```

```
      "You do not have permission to view this page",
      ""
   );
}
```

Configuring the Virtual Server

Once you have a rule with the appropriate settings configured, you must assign it to the virtual server on which you want to enable authentication:

1. Go to the **Services > Virtual Servers** section of the Admin UI and select the virtual server on which you want to enable authentication.
2. Click the **Rules** section.
3. Under **Request Rules**, select your authentication rule and click '*Add Rule*'.

CHAPTER 10 Java Extensions

This chapter gives an overview of the Java support available. Java can be called from TrafficScript, which allows for more powerful routing and easier traffic management.

For a more in-depth description, refer to the *Brocade Virtual Traffic Manager: Java Development Guide*.

Introduction to Java

Java is an object-oriented and platform-independent development language that has a large community of developers, libraries and applications. The Traffic Manager allows the use of Java extensions in TrafficScript, offering greater flexibility in traffic manipulation. These Java extensions can then be invoked from TrafficScript rules.

The Traffic Manager Java API is a way for customers and Internet Service Providers to extend the capabilities of the Traffic Manager.

Some examples of functionality available using the Java API in TrafficScript are:

- **Content processing -** Improved XML/HTML processing using specialized Java libraries.
- **Additional libraries -** ISV libraries supplied as a value-add solution, operating as 'pluggable' extensions to the Traffic Manager to add new features.
- **Authentication -** Using RADIUS/TACACS+/LDAP etc., without the need to use an external Web server.

Invoking a Java Extension

Java extensions are invoked from TrafficScript or RuleBuilder rules.

Whenever a Java extension is imported into the Traffic Manager, a basic RuleBuilder rule is created with the same name as the Java Extension. The rule invokes the extension, allowing it to inspect, modify and route the traffic. You can add this basic rule to the list of request or response rules run by a virtual server.

In many cases, a particular Java extension need not be called for every single request or response; for example, it processes only requests or responses of a particular type. The Java Extension can be called conditionally by a RuleBuilder or TrafficScript rule:

```
if( http.getPath() == "/serverstatus" ) {
```

```
    java.run( "ServerStatus" );
}
```

Configuring the Traffic Manager to Use Java

This section introduces the process of creating, configuring and running Java Extensions on the Traffic Manager. The Java Development Guide provides a much more complete description of this process.

Requirements

In order to use Java Extensions with the Traffic Manager, you must install the Sun Java run-time environment (JRE) version 1.5 or later. Previous versions are not supported by the Traffic Manager.

Traffic Manager virtual/cloud instances include the necessary Java software to run Java Extensions.

Compiling a Java Extension

To compile Java Extensions for the Traffic Manager you will need the following resources:

- Java Development Kit (JDK), which contains the Java compiler. The compiler can be downloaded from http://java.sun.com.
- Java Servlet API library: This can be found in `$ZEUSHOME/zxtm/lib/servlet.jar` or downloaded from the link in the **Java Extensions catalog**.
- Traffic Manager Java Extensions API library: This can be found in `$ZEUSHOME/zxtm/lib/zxtm-servlet.jar` or downloaded from the link in the **Java Extensions catalog**.

To compile a Java Extension, run the command:

```
$ javac -cp servlet.jar:zxtm-servlet.jar MyServlet.java
```

This will create a class file called *MyServlet.class.* It is assumed that you have copied *servlet.jar* and *zxtm-servlet.jar* to the directory you are compiling from (if not, you have to specify their paths as well).

You can also package up a Java extension along with any other needed classes in a single jar file. The Traffic Manager will automatically search jar files for Java extensions to use.

Loading Java Extensions onto the Traffic Manager

Java Extension class and jar files need to be added to the Traffic Manager's **Java Extensions Catalog**.

To upload the Extension, go to the **Catalog > Java Extensions** page and specify your class or jar file in the Upload section. If your extension depends on other Java jar files that are not included in the Java distribution, you should upload those into the catalog too.

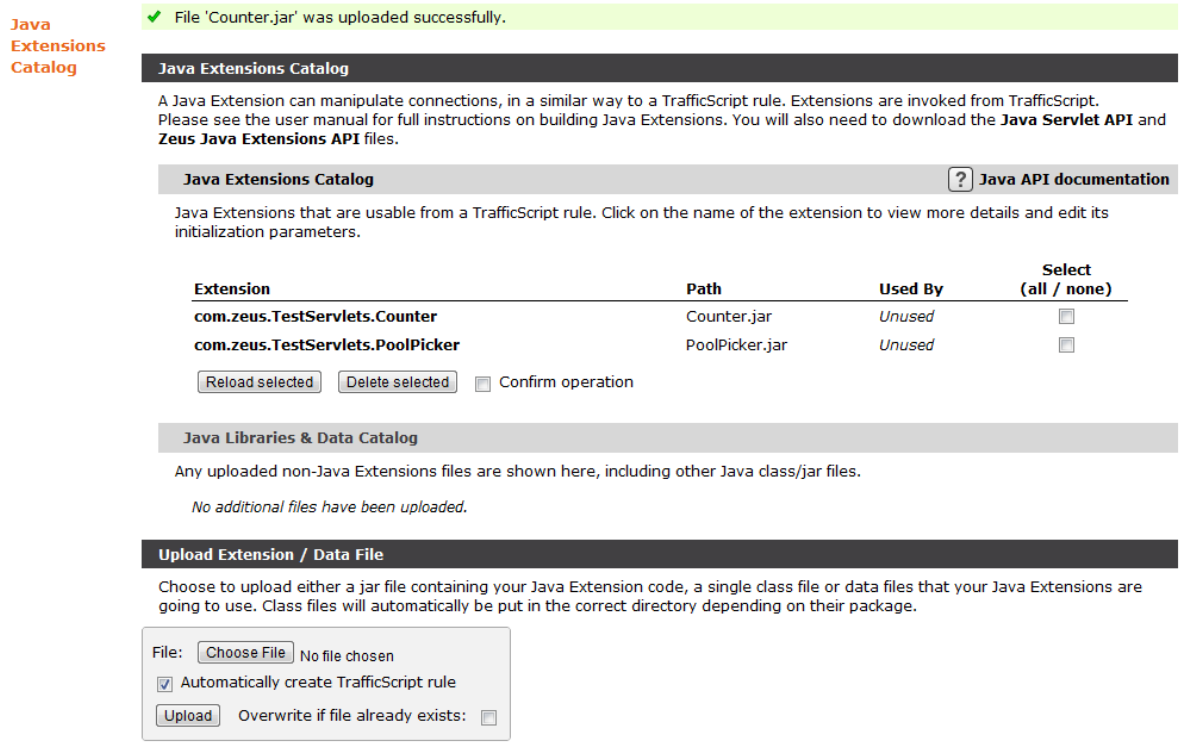

Java Extensions Catalog
File 'Counter.jar' was uploaded successfully.
Java Extensions Catalog
A Java Extension can manipulate connections, in a similar way to a TrafficScript rule. Extensions are invoked from TrafficScript. Please see the user manual for full instructions on building Java Extensions. You will also need to download the Java Servlet API and Zeus Java Extensions API files.
Java Extensions Catalog
Java API documentation
Java Extensions that are usable from a TrafficScript rule. Click on the name of the extension to view more details and edit its initialization parameters.
Extension
Path
Used By
Select (all / none)
com.zeus.TestServlets.Counter
Counter.jar
Unused
com.zeus.TestServlets.PoolPicker
PoolPicker.jar
Unused
Reload selected
Delete selected
Confirm operation
Java Libraries & Data Catalog
Any uploaded non-Java Extensions files are shown here, including other Java class/jar files.
No additional files have been uploaded.
Upload Extension / Data File
Choose to upload either a jar file containing your Java Extension code, a single class file or data files that your Java Extensions are going to use. Class files will automatically be put in the correct directory depending on their package.
File:
Choose File
No file chosen
Automatically create TrafficScript rule
Upload
Overwrite if file already exists:

Fig 30. Java Extensions Catalog page

Configuring the Traffic Manager's Java Extension Runner

The Traffic Manager includes a Java helper application called the Java Extension Runner that hosts the Java extensions. You can control how the Traffic Manager initializes and runs this helper application using the settings in the **System->Global Settings->Java Extension Runner** section of the Admin Interface.

To specify a Java executable, set the "java!command" field in the **[]** section of the interface to the name of the executable (and the path if it is not the default), along with any options the JRE should be run with.

- **java!command**: The path to the java executable, including any command-line arguments
- **java!classpath**: Colon-separated list of folders where the Java classes are located.
- **java!lib**: Specifies a folder to search automatically for libraries used by your Java extensions. Note that the Traffic Manager will not load any Java extensions from this directory.
- **java!max_conns**: Defines the maximum number of simultaneous Java requests allowed. Additional requests will be queued and will not be processed until the former ones have been completed. This setting is per CPU core; for multicore processors you will have to multiply this setting by the actual number of processors.
- **java!session_age**: Value in seconds, defining a timeout to maintain a java session.

To check the setup, press the **Diagnose** button, go to the **Cluster diagnosis** tab, and verify that the "Java extensions" line is green. The Traffic Manager should now be ready to run Java extensions.

CHAPTER 11 Protocol Support

The Traffic Manager load-balances and processes application traffic that is enclosed in TCP connections or UDP datagrams. The majority of features, including TrafficScript, load balancing, session persistence, bandwidth management etc can be applied equally to all protocol types.

Where appropriate, additional TrafficScript functions specific to a particular protocol are included, making management and control of traffic in that protocol easier.

Basic TCP Protocols

'Generic Server First', 'Generic Client First' and 'Generic Streaming' protocols are the most basic L7 protocol types that the Traffic Manager can use. They are useful when managing custom protocols or simple TCP connections because they do not expect the traffic to conform to any specific format.

Server-First Protocols

Server-first is the simplest TCP protocol type. When the Traffic Manager receives a connection from a client, it immediately runs any TrafficScript request rules, then immediately connects to a back-end server. It then listens on both connections (client-side and server-side) and passes data from one side to the other as it comes.

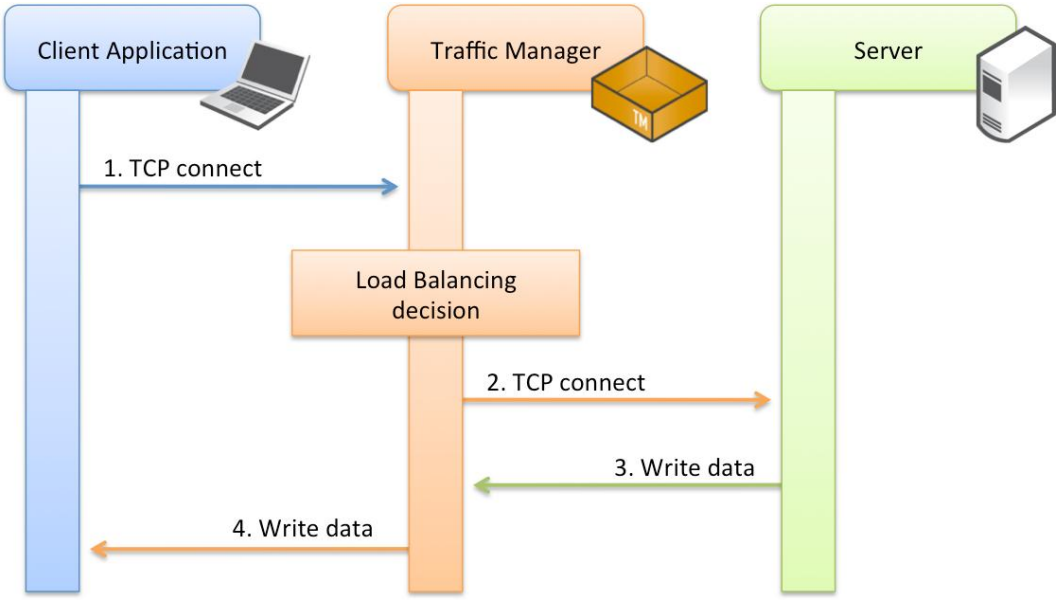


Fig 31. Server-first protocol load balancing

Server-first protocols include one-shot protocols such as Time (where the server writes the response data, then immediately closes the connection), or complex

protocols like MySQL (where the server opens the dialog with a password challenge).

The virtual server protocol type 'Generic Server First' is most suitable for protocols where the server 'speaks' first. In practice, the design of most protocols means that 'Generic Client First' is more appropriate.

Client-First Protocols

Client-first is a modification of server-first, appropriate for protocols where the client connects and sends a request before the server replies. You can use the "Generic Client First" protocol type to inspect a client's request and select a server pool that should be used to select the server node to be used.

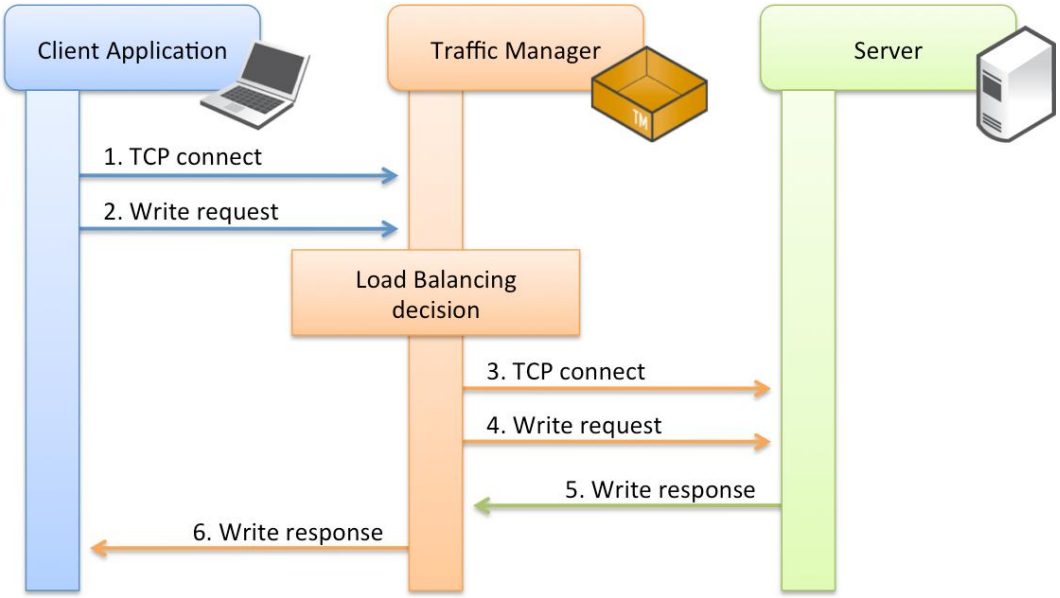


Fig 32. Client-first protocol load balancing

The Traffic Manager is only alerted when a client connection has been established and data has been received. The Traffic Manager then proceeds in the style of server-first, i.e. runs TrafficScript rules, connects to the back end and relays data back and forth.

Rules Processing in Detail

The Traffic Manager alternates between running TrafficScript request and response rules. It will run the request rules when the first data from the client received; the request rules may block if they use functions like `request.getLine()` or `request.get()` to read further data from the client.

Once the request rules have completed, the Traffic Manager will forward any further data it receives from the client on to the server. The Traffic Manager will run the

response rules as soon as it receives any data (i.e. a “response”) from the server; these response rules may also block.

The Traffic Manager will then continue to forward any further data from the server to the client; the Traffic Manager will wait for any data from the client and run the Request rules again; in this manner, the Traffic Manager switches between waiting for request data to run request rules, and waiting for response data in order to run response rules.

Note that rules that are configured as “Run Once” rather than “Run Every” will only be run on the first server or client data, not on subsequent iterations.

Server-First with "Server Banner"

Server-first with a 'server banner' is a different optimization for ‘server-first’ to cater for servers which broadcast a banner on connect, such as SMTP. ‘Server-first with server banner’ allows you to inspect and make a load-balancing decision based on the client’s request.

When a client connects, the Traffic Manager immediately writes the configured 'server-first banner' to the client, then proceeds as a regular client-first connection. In addition, the Traffic Manager slurps and discards the first line or data (terminated by a newline) that the server sends.

Once again, you can use TrafficScript rules to inspect the client's request data before making your load-balancing decision.

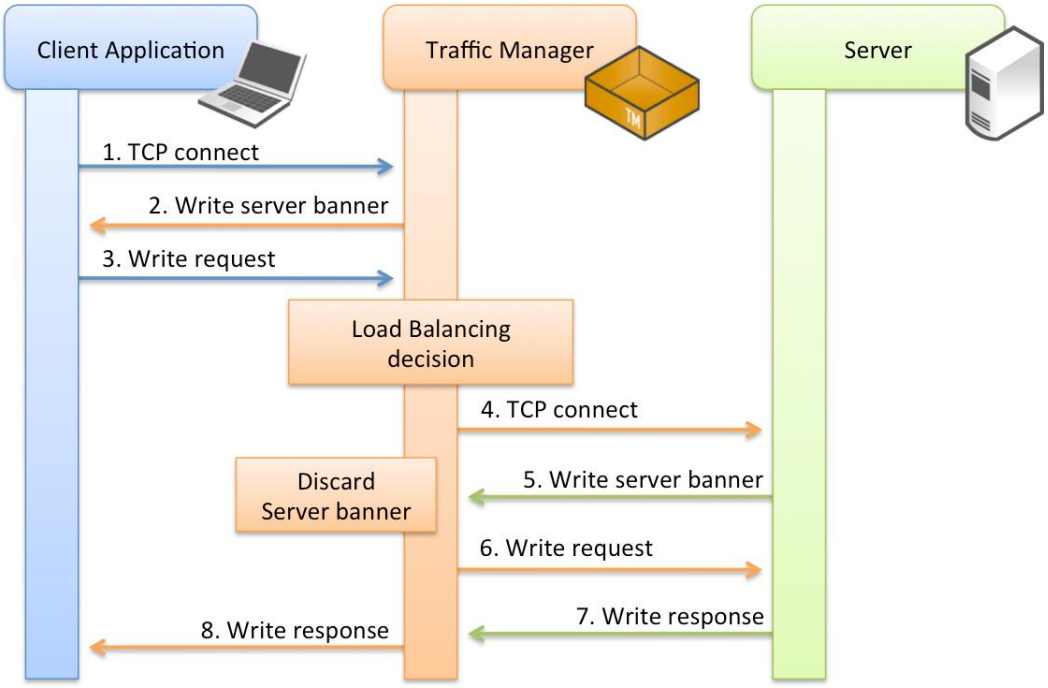


Fig 33. Server-first load balancing with Server Banner

To configure this behavior, select the 'Generic Server First' protocol type and configure a banner message in the **Connection Management** settings in the virtual server configuration.

Generic Streaming Protocols

If you are using a protocol that does not require the server to respond to every message that the client sends, or if you need extra flexibility when processing data in TrafficScript, you can choose the 'Generic streaming' option. This server type allows either the client or server to send the first message when a connection is established and also allows TrafficScript rules to be invoked whenever data is received on the connection.

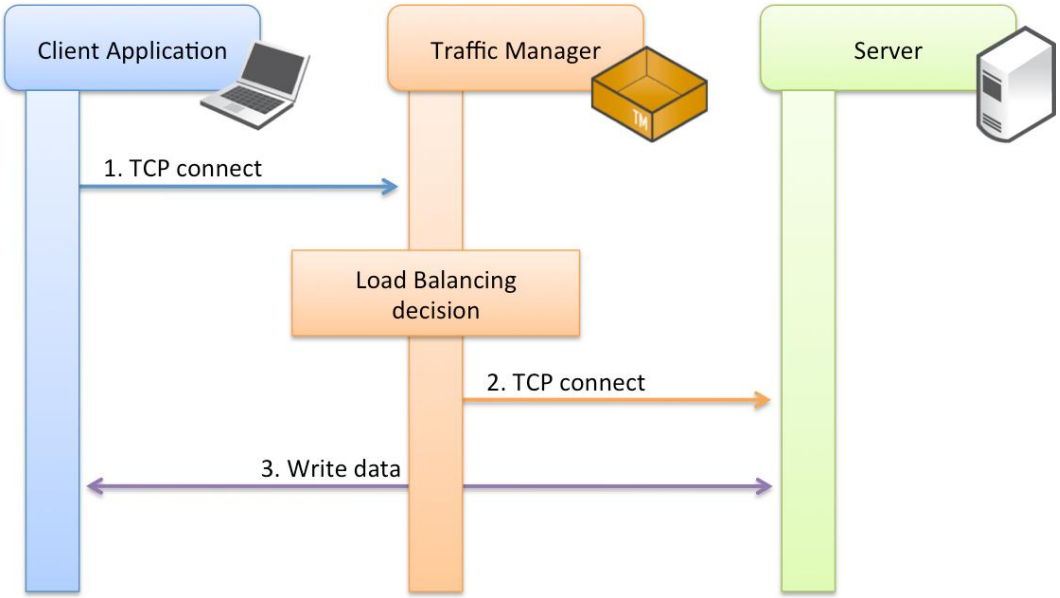


Fig 34. Load balancing with generic streaming protocols

This option is for protocols where there is no request-response semantic. Either side of the connection can write the first message, with no response being necessarily required or expected. TrafficScript request rules are run whenever the client writes data on the connection, and similarly response rules are run every time the server writes data on the connection.

HTTP

The Traffic Manager's HTTP virtual server protocol type contains a number of optimizations and specializations for HTTP traffic.

- The Traffic Manager manages client-side and server-side connections independently, re-using keepalive connections on the server side whenever possible to reduce the number of established and new TCP connections to the

server. This minimizes the number of concurrent connections the servers need to handle, and brings big performance and capacity gains.

Note that features such as HTTP POSTs and missing content lengths can make keepalives unsafe to use; the Traffic Manager detects when this occurs and creates new connections appropriately. NTLM authentication specifically requires that keepalives are enabled.

- The Traffic Manager conceals the use of keepalives and pipelining from the administrator, so that traffic management rules need only concentrate on the simple request-response nature of an HTTP transaction. Every HTTP transactions is processed and handled independently, regardless of whether or not it is in a keepalive connection.
- The Traffic Manager automatically handles HTTP encodings, such as gzip and deflate content compression and chunked transfer encoding. For example, if you inspect an HTTP response using the `http.getResponseBody()` TrafficScript function or from within a Java Extension, the Traffic Manager will automatically de-chunk and uncompress the response so that it can be easily manipulated.
- The Traffic Manager includes a large set of specialized HTTP TrafficScript functions that make it easy to process HTTP requests and responses. For example, functions are provided to manipulate HTTP cookies, read HTTP headers (such as the host header) and process URLs without having to understand the underlying encodings, variations in format and protocol details that complicate these tasks.

SSL

When you configure the Traffic Manager to manage traffic using the SSL protocol, the Traffic Manager does not automatically decrypt the traffic. The SSL protocol type is used for SSL pass-through, without modification.

You can use the 'SSL Session ID' persistence type to load-balance SSL connections across a pool, ensuring that connections with the same SSL session ID are sent to the same back-end server. This is an important consideration because SSL servers do not generally share SSL session IDs with each other. So if a client attempts to re-use an SSL session ID, but is directed to a new server, it would then need to renegotiate its SSL session. This is compute-expensive and will add load to your SSL server farm. In fact, without SSL session ID persistence, adding more SSL servers to your cluster can reduce performance because SSL sessions need to be renegotiated more frequently.

Note: if you want to decrypt SSL traffic and process it using the internal protocol (such as HTTP), you should configure your virtual server to use the internal protocol, and enable SSL decryption in the virtual server configuration settings (see the *SSL Decryption* section of CHAPTER 4, and CHAPTER 13, for details about configuring SSL).

Protecting the SSL Handshake

It is possible to place a limit on the size of the peer handshake message clients attempt when negotiating an SSL connection. This can help to protect against a potential Denial-of-Service (DoS) attack.

In the Admin UI, navigate to the **System > Global Settings** page. In the *SSL Configuration* section, you will find a config key **ssl!max_handshake_message_size**.

The value of this setting determines the maximum size of SSL handshake messages accepted for SSL connections. Any handshake message indicating a size larger than this setting will cause the SSL connection to fail and will be logged as a potential DoS attack in the event log. The default value of 10 KiB should be able to accept a chain of 5 certificates, which should be more than sufficient for typical SSL set-ups. Specifying a value of 0 denotes that there is no limit to the size of SSL handshake messages handled by the Traffic Manager.

SSL Connection Renegotiation Protection

Both SSL 3.0 and TLS (Transport Layer Security) protocols allow either the client or server to initiate renegotiation of the secure connection. This might be to establish new cryptographic parameters by which the connection will be governed.

During renegotiation, the secure connection becomes potentially vulnerable to interception by an attacker who might seek to inject traffic as a prefix to the client's interaction with the server. This mechanism, known as a Man-in-the-Middle (MitM) attack, can be used to set up a new client connection from the attacker, fooling the target server into believing that the initial data transmitted by the attacker is from the same entity as the original client.

Furthermore, renegotiation can be used to carry out a Denial of Service (DoS) attack because it poses a much higher load on the server than on the client. A client could simply request a re-handshake at a very high rate causing a very high computational burden on the server.

RFC-5746 (http://tools.ietf.org/html/rfc5746) introduces a TLS extension that prevents the MitM attack by cryptographically tying renegotiations to the client that is performing it. However, the introduction of this extension does not necessarily prevent or protect from a DoS attack.

The Traffic Manager provides a configuration setting (`ssl!ssl3_allow_rehandshake`) that allows you to select how a potential renegotiation handshake should be handled. You can find it on the **System > Global Settings** page of the Admin UI, within the **SSL Configuration** section. The available options are:

Setting	Description
Always allow	Renegotiation is always allowed, even if the client lacks support for RFC-5746, but without any form of protection against intervention by a third party. This setting is NOT RECOMMENDED, and could leave your system open to MitM or DoS attacks.
Allow safe re-handshakes [default setting]	This includes the case in which the client supports RFC-5746 and the case that no data has yet been transmitted over the connection. Since the aforementioned attack consists of pre-pending plain text, if no data has yet been received, the re-handshake is safe. This is of limited use as from the client's perspective, the re-handshake is actually the first handshake, so the client cannot know whether the connection is under attack. This option is therefore mostly provided for backward compatibility with clients that have yet to be upgraded to support RFC-5746. Note that since this setting allows re-handshakes it still leaves an installation potentially open to the DoS attack mentioned above.
Only if client uses RFC 5746 (Secure Renegotiation Extension)	Renegotiation handshakes are only allowed if the client supports *Secure Renegotiation* as defined in RFC-5746. Note that since this setting allows re-handshakes it still leaves an installation potentially open to the DoS attack mentioned above.

Setting	Description
Never allow	Reject all attempts at a renegotiation handshake. This is the most secure method as it protects against both MitM and DoS attacks. However, service failure is possible if, for example, some aspect of your installation depends on client-side renegotiation.

If SSL 3.0/TLS re-handshakes are allowed, you can use the `ssl!ssl3_min_rehandshake_interval` setting to define the minimum time interval (in milliseconds) permitted between handshakes on a single SSL 3.0/TLS connection. The time interval starts when the opening handshake is initiated and is reset at the beginning of every re-handshake. To disable the minimum interval for handshakes, use a setting of 0 (zero).

If the Traffic Manager acts as an SSL server (`ssl_decrypt` is set to "Yes" in a virtual server) and a renegotiation is triggered by the Traffic Manager itself via the TrafficScript function **ssl.requirecert()**, the resulting renegotiation will not be subject to the rate limit.

SMTP (Simple Mail Transport Protocol)

SMTP is a server-first protocol, but with one additional capability. For improved security, an SSL client can request that an SMTP connection is encrypted using SSL (also known as TLS).

If you configure a virtual server to use the SMTP protocol, it will behave just like a regular Generic Server First protocol (see the *Server-First Protocols* section of CHAPTER 11), and you will probably want to configure a server first banner (see the *Server-First with "Server Banner"* section of CHAPTER 11).

However, if you enable SSL for an SMTP virtual server, you will find an additional setting named **ssl!expect_starttls** in the **Virtual Server > SSL** configuration page:

- If **expect_starttls** is set to 'no', the virtual server will decrypt SSL traffic in the normal way, i.e., expect traffic to be SSL-encrypted at the very beginning of the connections.
- If **expect_starttls** is set to 'yes', the Traffic Manager will process the traffic in plain text, as if it were unencrypted. The Traffic Manager will watch for a 'STARTTLS' SMTP command from the client and will then initiate an SSL handshake to upgrade the client's connection to SSL.

The Traffic Manager will forward all traffic on to the back-end SMTP servers unencrypted (in plaintext) unless you enable SSL encryption in the pool configuration. If you do this, the Traffic Manager will encrypt the connection from the outset, rather than using the STARTTLS SMTP command.

FTP

The FTP protocol is used to transfer files from client to server. An FTP session consists of:

- A control connection, initiated by the client to port 21 (typically) on the remote server; this connection is received and proxied by the Traffic Manager.
- One or more data connections that are created on-the-fly as a result of commands sent down the control connection; these connections may be initiated by the server to the client ("active mode") or by the client to the server ("passive mode").

The Traffic Manager's 'FTP' virtual server type includes a full FTP proxy that intercepts and manages the requests to create additional data connections. The Traffic Manager proxies the data connections just as it proxies the control connection, supporting both the active and passive modes of data transfer.

The proxy fully supports all commonly used FTP data connection commands, including the long (RFC 1639) and extended (RFC 2428) versions. The proxy does not support the RFC 4217 encryption control statements ('AUTH TLS', 'AUTH SSL', 'CCC'), and removes these commands from the control stream.

Note: **consider source port** should not be used with any Traffic IP Addresses that are used by FTP virtual servers (see the *Creating a Traffic IP Group* section of CHAPTER 6).

FTP Security

FTP is not a secure protocol. Passwords are not used for 'anonymous' file transfers which are common across the public Internet, but where passwords are needed, they are sent in plain text and vulnerable to eavesdropping. If this is of concern, alternative protocols such as SCP, WebDAV or SSL-wrapped FTP (see the *SSL-Wrapped FTP* section of this chapter) must be used.

A second concern is the ease with which an eavesdropper can snoop on the dialog that prepares the client and server for a data connection, and can then step in and initiate the connection to the listening party.

For example, when an Active FTP connection is set up, the client informs the server of the local client port that the server should connect to in order to establish the data

connection. An eavesdropper could step in and connect to the client's local port before the server does, and then capture any data the client sends (or return a fake file to the client).

The FTP protocol does not have any built-in security measures to authenticate connections and to prevent this from happening, but many modern FTP clients and servers will check the source IP address of incoming connections to verify that they originate from the remote FTP agent they are communicating with. This behavior is enabled by default in the Traffic Manager – go to **Virtual Server > Connection Management** and examine the settings in the **FTP-Specific Settings** section of that page:

- **ftp_force_client_secure** (default 'yes'): Verify that all passive data connections originate from the client's IP address.
- **ftp_force_server_secure** (default 'yes'): Verify that all active data connections originate from the server's IP address.
- **port_range**: Specify the range of destination ports used for data connections; these ports may need to be permitted through intervening firewalls so that, for example, clients can make passive connections to the Traffic Manager.

FTP Source Ports

The FTP specification recommends that FTP server data connections should use a source port one below the FTP service port. The typical FTP service port is port 21, so the data source port is recommended to be port 20.

In practice, the vast majority of FTP clients and servers ignore this recommendation because it has security implications and does not provide any additional assurance or authentication. By default, the Traffic Manager will select high port numbers on a random basis for FTP data connections that originate from the Traffic Manager.

Specifying the Source Port

You can use the **ftp_data_source_port** virtual server setting (in the **Connection Management > FTP Settings** section) to specify a source port that must be used. This is often required when an upstream firewall is used to block outgoing connections, other than those permitted with an explicit policy (for example, source port and IP).

FTP software in a Unix/Linux environment requires certain permissions to issue requests from low ports (lower then 1024), such as port 20. By default, the Traffic Manager drops these permissions early (principle of least privilege), so attempts to configure **ftp_data_source_port** to a low value will fail. If you wish to use low ports, enable the setting **ftp_bind_data_low** in the **System Settings** section of the **Global**

Settings page; this will cause the Traffic Manager relinquish fewer permissions so that it retains the ability to bind to low ports.

On modern Linux platforms only the privilege to bind to low ports is kept, while on Unix variants full root privileges will be retained.

SSL-Wrapped FTP (FTPS)

The Traffic Manager supports the pre-RFC 4217 use of FTPS (referred to as 'Implicit FTPS' or 'SSL-wrapped FTP'). In this implementation, the control port is encrypted from the very beginning; the data connections are generally encrypted but can operate in plaintext if desired.

The virtual server protocol type for SSL-wrapped FTP is 'FTP'; the virtual server and pool can be independently configured to decrypt or encrypt the control channel communications.

Controlling the Control Channel

- **ssl_decrypt**: Enable the ssl_decrypt setting on the **Virtual Server > SSL Decryption** page to enable SSL decryption for client-side connections. Refer to the *Setting Up SSL Decryption* section of CHAPTER 13 for more information on configuring SSL decryption.
- **ssl_encrypt**: Enable the ssl_encrypt setting on the **Pool > SSL Settings** page (see the *SSL Encryption* section of CHAPTER 13) to encrypt the control connection before sending it to a back-end server.

The Traffic Manager's FTP proxy disables the encryption control statements ('AUTH TLS', 'AUTH SSL', 'CCC') in RFC 4217 so that they cannot interfere with the established SSL-wrapped FTP session.

Controlling the Data Channel

The Traffic Manager can decrypt the data channel or leave it unmodified: this behavior is controlled by the **Virtual Server > SSL Decryption** setting ftp!ssl_data.

- If enabled, ftp!ssl_data will cause the virtual server to decrypt all data connections using the same public certificate as is used for the control port. Note that the FTP virtual server must also be configured to decrypt SSL with ssl_decrypt.
- If disabled, the virtual server will pass all data through unmodified. If the back-end server wishes to use SSL on the data connection, the client can negotiate directly with the back-end server, with the Traffic Manager as an intermediate proxy.

Note that on the server side, the Traffic Manager will encrypt the data connection if either:

- The virtual server setting ssl_decrypt is disabled (the virtual server is not decrypting SSL), and the pool setting ssl_encrypt is enabled (the pool is encrypting the server-side of the data channel). In this case, data communications with the clients will be in plaintext.
- The virtual server settings ssl_decrypt and ftp!ssl_data are both enabled (the virtual server is decrypting SSL), and the pool setting ssl_encrypt is enabled (the pool is encrypting the server-side of the data channel). In this case, data communications with the clients will be encrypted.

Use Cases for SSL-Wrapped FTP

Use Case 1: Servers and Clients Support SSL-Wrapped FTP

In this use case, all parties support and use SSL-wrapped FTP and the Traffic Manager acts as a load-balancing proxy, connecting clients to servers. The Traffic Manager decrypts and re-encrypts the control connection internally so that it can be inspected, and the Traffic Manager can optionally decrypt and re-encrypt the data connection.

- Enable SSL decryption (virtual server: ssl_decrypt) and SSL encryption (pool: ssl_encrypt) so that the Traffic Manager can inspect and proxy the control channel, and create data connections dynamically.
- Optional: enable ftp!ssl_data. If disabled, clients and servers can negotiate whether or not to use encryption on the data connections. If enabled, the Traffic Manager will require clients to use SSL for data transfers.

Note: Enabling ftp!ssl_data brings two benefits: First, certificate management is easier as all connections are encrypted and authenticated against the Traffic Manager virtual server certificate. Second, the Traffic Manager will force all clients to use SSL for their data connections.

Without **ftp!ssl_data**, you would need to synchronize certificates across the Traffic Manager and the back-end servers, and you could not prevent clients and servers from using plaintext (unencrypted) data connections. However, these benefits come at the cost of additional SSL processing on the Traffic Manager system.

Use Case 2: Clients Support SSL-Wrapped FTP; Servers Do Not

This configuration can be used with FTP servers that do not support SSL or in situations where the Traffic Manager and FTP servers are in close proximity and there is no need to secure the network connections between them:

- Enable SSL decryption (virtual server: ssl_decrypt), but do not enable SSL encryption (pool: ssl_encrypt). The Traffic Manager will decrypt SSL-wrapped FTP from the client, and pass it in plain-text to the FTP servers in the pool.
- Recommended: enable ftp!ssl_data. If disabled, clients and servers will always use plaintext (unencrypted) connections for data transfer. If enabled, the Traffic Manager will force clients to use SSL for data transfers.

This use case illustrates an easy way to upgrade your FTP services to SSL-wrapped FTP without requiring any server modifications.

Real-Time Streaming Protocol

The Real-Time Streaming Protocol (RTSP) allows a client to establish and control either a single or several time-synchronized streams of continuous media, such as audio and video servers. Applications based on RTSP act as a video-like remote control panel for multimedia servers. Some servers also use the Session Initiation Protocol (SIP) combined with RTSP in the same conference.

RTSP is used by client applications such as QuickTime, MPlayer, VLC, Windows Media Player and RealPlayer.

The Traffic Manager is able to recognize and manage RTSP traffic, allowing users to load balance a pool of media servers that handle this type of traffic.

The figure below shows the usual layout of a RTSP connection. An RTSP client will establish a TCP connection to an RTSP server, via the Traffic Manager, over which RTSP requests will be issued. This is the control channel. The media data itself may be sent independently over UDP (the most common scenario), or multiplexed over TCP using the same connection.

Alternatively HTTP may also be used for data delivery. Note that the client and server negotiate during the setup if the underlying transport protocol to be used is UDP or TCP; this configuration is not done from the Traffic Manager.

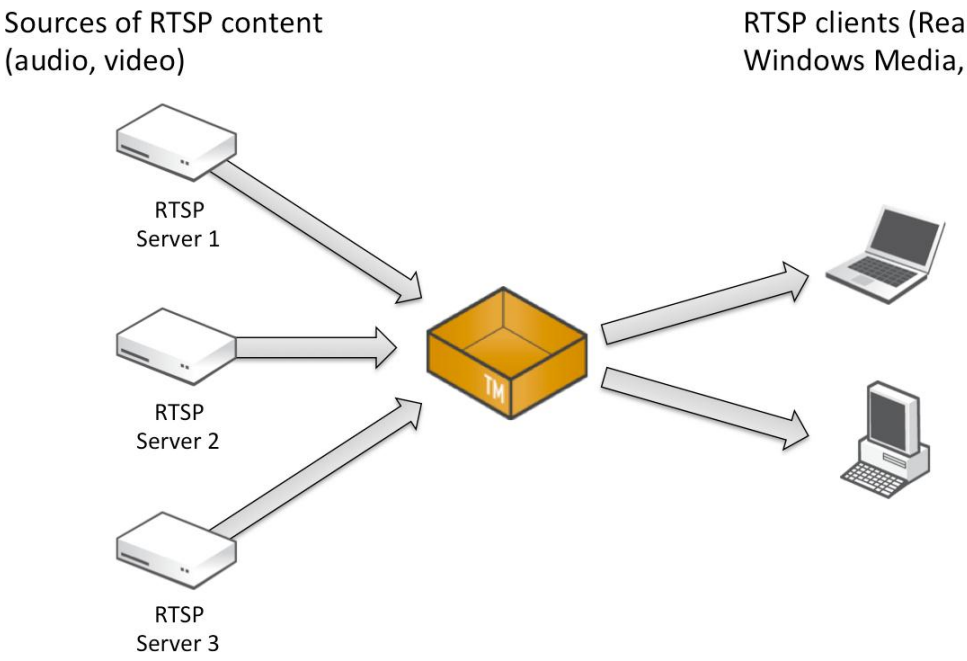


Fig 35. A Traffic Manager acting as a load balancer for standard RTSP content traffic

RTSP itself only *controls* the transport of the media. The protocols involved in a full media transaction may include the following:

- **RTP**: Real-time Transport Protocol, used to transport the data.
- **RTCP**: Real Time Control Protocol, used to provide extra information for an RTP stream.
- **RDT**: Real Data Transport, a proprietary data delivery format from Real Networks that transports the data stream.

From a functional point of view, the use of a Traffic Manager for the handling of RTSP:

- Reduces the stress on the servers and allows load balancing across a pool of RTSP servers.
- Customizes and improves the performance, by using TrafficScript rules.
- Provides session persistence in the case of a server failure.

Setting Up an RTSP Service

RTSP runs over port 554 by default. To manage an RTSP service with the Traffic Manager, create a virtual server listening on port 554 (or an alternative port if appropriate), and select the protocol type 'RTSP'.

Your virtual server will need to use additional ports to stream content over UDP. If you need to specify the range of ports which will be used, use the **Connection**

Management settings for the RTSP virtual server. In the **RTSP-Specific Settings** section you may modify the port range. It is recommended that you enter the largest range that you can.

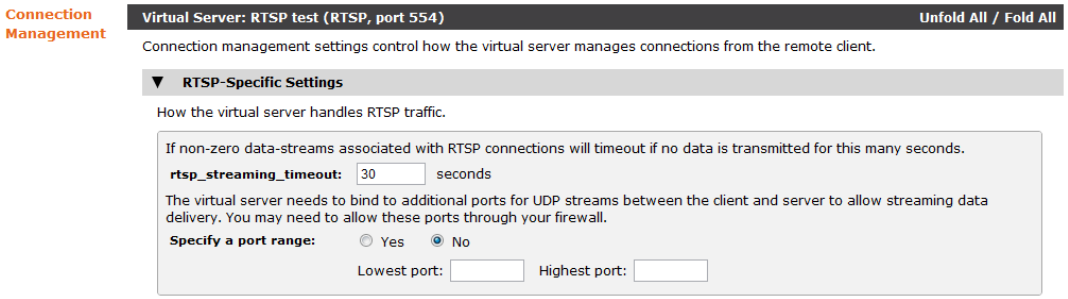

Connection Management
Virtual Server: RTSP test (RTSP, port 554)
Unfold All / Fold All
Connection management settings control how the virtual server manages connections from the remote client.
RTSP-Specific Settings
How the virtual server handles RTSP traffic.
If non-zero data-streams associated with RTSP connections will timeout if no data is transmitted for this many seconds.
rtsp_streaming_timeout: 30 seconds
The virtual server needs to bind to additional ports for UDP streams between the client and server to allow streaming data delivery. You may need to allow these ports through your firewall.
Specify a port range: Yes No
Lowest port: Highest port:

Fig 36. Configuring the port range for RTSP

The total number of ports needed depends on the transport method and specific servers used, the specific player used by the clients and the format of the files to be streamed. For example, in a system with 100 active clients, up to 8 ports per client might be needed, giving us a total number of 800 ports.

Session Initiation Protocol

Session Initiation Protocol (SIP) is an application-layer control protocol that can initiate, control, modify or terminate sessions with one or more participants. SIP is typically used for telephone and video calls, streaming multimedia distribution and conferencing. The SIP protocol offers the following features:

- User location
- User availability
- User capabilities
- Session setup
- Session management

Features of SIP

This section outlines the features of SIP in more detail.

- **User Location:** When a user activates a SIP device (e.g. a SIP phone), it registers its current location with a SIP server. The user can then be contacted at the server's domain. Each user can have several devices in different locations registered with a server at the same time.

When someone attempts to contact the user, the message will arrive at the server. The server will then choose one or several of the user's registered locations to forward the message to.

- **User Availability:** When a SIP device receives a message it can respond in several different ways. The response might be determined by user input – for example the user answering an incoming call. Alternatively, the response might be automatically generated, for example if the device is already in use a 'Busy' response might be returned. The server will decide what to do with this response - it might send it back to the caller or it might try sending the original message to a different location.

  If the user has no devices registered with the server then they are considered to be unavailable.

- **User Capabilities:** SIP devices are able to query other SIP devices to learn what capabilities they have. Communication can then take place using the features that both devices support.

- **Session Setup:** SIP can be used with the Session Description Protocol to establish a direct communication channel between two devices. This communication channel is then used to transmit session data, such as voice data in the case of a telephone call, while the SIP channel is used to send control information, such as a request to disconnect the call. Session data is typically transmitted using the Real-Time Transport Protocol (RTP).

- **Session management:** SIP controls termination of sessions and transfer of sessions between different devices. It is also possible to change the parameters of an established session using SIP.

SIP supports both IPv4 and IPv6, and operates over TCP and UDP.

The Traffic Manager and the SIP Protocol

The Traffic Manager load-balances SIP traffic between SIP proxy servers, creating a SIP infrastructure with high availability, reliability and extensibility. The figure below shows an example of a SIP network that includes a Traffic Manager. The SIP phones are typically located at the users' individual locations, while the Traffic Manager, the SIP proxy servers and the location database are owned by a SIP service provider.

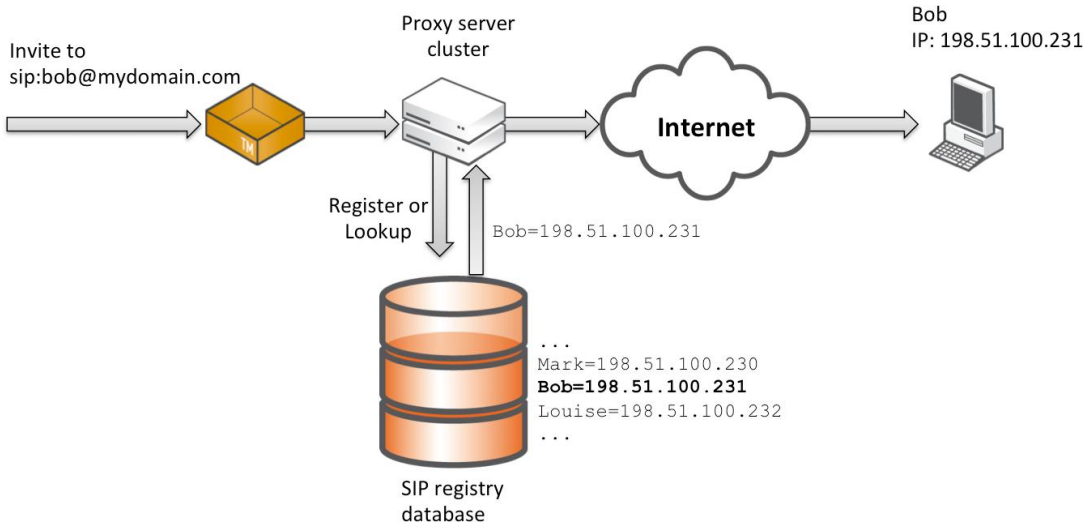


Fig 37. Detail of how an IP phone call using SIP would work with a Traffic Manager

When working with SIP traffic, the Traffic Manager provides the following features:

- Load-balance SIP requests between SIP proxy servers with customizable load-balancing algorithms.
- Maintain session persistence so all requests that are part of the same session will be sent to the same proxy server (by default, the Traffic Manager persists on the SIP "Call-ID" header).
- Modify SIP request and response packets as they pass through the Traffic Manager using RuleBuilder and TrafficScript rules.
- Issue responses to incoming requests without sending them to a proxy server.
- Monitor the health of the proxy servers in use and direct traffic away from them when they go down.
- Act as a gateway for all SIP data and session data when all clients are in a private network.
- Protect against Denial of Service attacks and malicious data.

Configuring the Proxy Servers to Support Traffic Management

The Traffic Manager is most effective when used in front of a series of SIP proxy servers, such as Oracle/Weblogic SIP server or SIP Express Router (SER). To work in this configuration, the proxy servers must recognize the domain clients use to contact the Traffic Manager as one they are responsible for. For example, if sip:user@example.com resolves to the IP of the Traffic Manager, then the proxy servers must be responsible for the domain example.com. The proxy server will then process the request instead of forwarding it to that domain, which would result in the request looping between the Traffic Manager and the proxy server.

Alternatively, if your proxy servers do not support domains, go to the **Virtual Servers > Edit > Connection Management** page of the Admin UI and enable the "rewrite_request_uri" option. Enabling this option will tell the Traffic Manager to replace the URI in each request it receives with the URI of the proxy server it sends the request to.

Setting Up a SIP Service on the Traffic Manager

A new SIP service can be created through the **Manage a new service** wizard. Specify a suitable name for the service, and choose either the SIP over TCP protocol or the SIP over UDP protocol. The default SIP port is 5060 for both TCP and UDP services. This wizard will set up a new SIP virtual server and pool with the details provided.

For your SIP virtual server to operate correctly, it must listen to all the IP addresses used to send or receive a SIP request. To do this, navigate to the **Virtual Servers > Edit > Basic Settings** page and locate the **Listening on** setting. Ensure that "All IP addresses" is selected (this is the default).

You can also use both the UDP and the TCP protocols on the same port by creating two virtual servers and assigning one of the protocols to each virtual server.

Important: When using the Traffic Manager in a SIP infrastructure, SIP requests are sent to the client on a different connection to the one used when they register with a SIP proxy server. As a result, the client's firewall must be configured to allow SIP requests through to the phone. Some firewalls will do this automatically, whereas others will need to be configured to forward requests on port 5060 to the user's phone.

SIP Operation Modes on the Traffic Manager

A SIP virtual server can run in three different operational modes: **SIP Routing, SIP Gateway** and **Full Gateway.** This setting affects how much involvement the Traffic Manager has over control information (SIP messages) and session data (typically RTP data in voice and video communication).

To define the operational mode of the SIP server, edit your server in the Traffic Manager admin interface, then click **Connection Management>SIP-specific settings.**

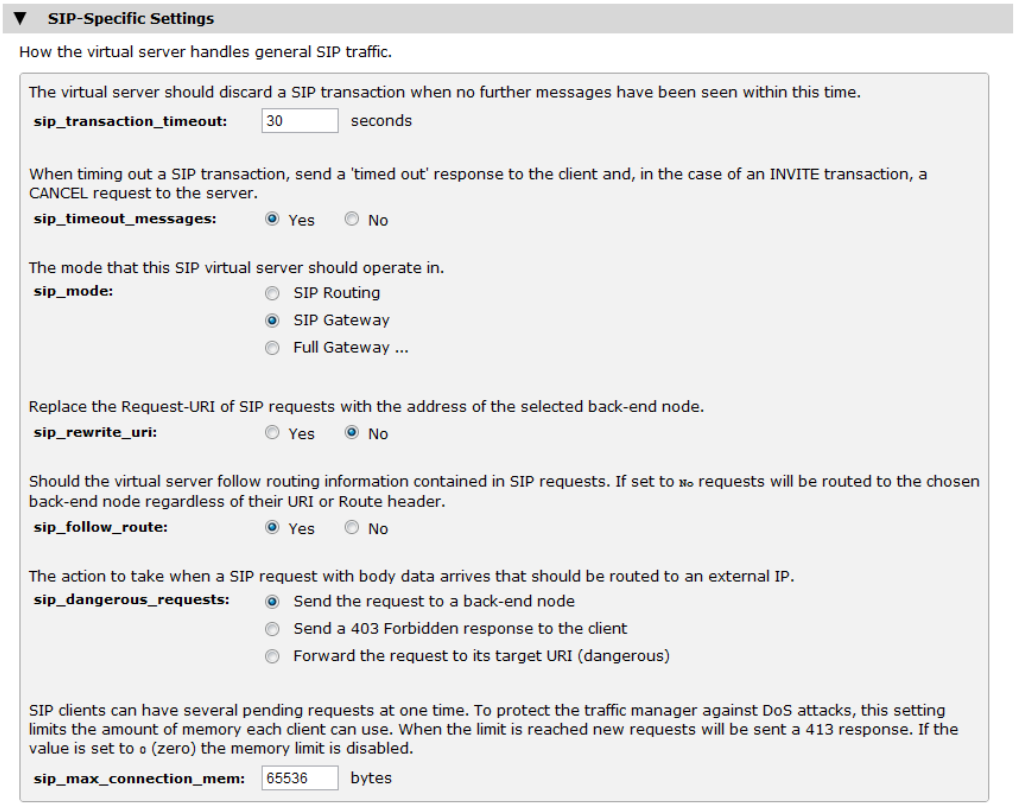

SIP-Specific Settings
How the virtual server handles general SIP traffic.
The virtual server should discard a SIP transaction when no further messages have been seen within this time.
sip_transaction_timeout: 30 seconds
When timing out a SIP transaction, send a 'timed out' response to the client and, in the case of an INVITE transaction, a CANCEL request to the server.
sip_timeout_messages: Yes No
The mode that this SIP virtual server should operate in.
sip_mode: SIP Routing
SIP Gateway
Full Gateway ...
Replace the Request-URI of SIP requests with the address of the selected back-end node.
sip_rewrite_uri: Yes No
Should the virtual server follow routing information contained in SIP requests. If set to No requests will be routed to the chosen back-end node regardless of their URI or Route header.
sip_follow_route: Yes No
The action to take when a SIP request with body data arrives that should be routed to an external IP.
sip_dangerous_requests: Send the request to a back-end node
Send a 403 Forbidden response to the client
Forward the request to its target URI (dangerous)
SIP clients can have several pending requests at one time. To protect the traffic manager against DoS attacks, this setting limits the amount of memory each client can use. When the limit is reached new requests will be sent a 413 response. If the value is set to 0 (zero) the memory limit is disabled.
sip_max_connection_mem: 65536 bytes

Fig 38. SIP virtual server settings

The different modes of operation are described below:

- **SIP Routing** - Select this option to load-balance SIP sessions between your SIP proxy servers. The first SIP request and all responses to it will pass through the Traffic Manager and can be inspected and manipulated by RuleBuilder and TrafficScript rules.

  The Traffic Manager will not add a Record-Route header field to the request. As a result, future requests that are part of the same session will not pass through the Traffic Manager but will instead go straight to the server that the original request was sent to.

  In a typical SIP session, this means the INVITE request and all responses to it will pass through the Traffic Manager. Any subsequent requests that manage this session, such as BYE requests, will not pass through the Traffic Manager.

- **SIP Gateway** - Select this option to inspect every SIP request that is part of a session. When a request passes through the Traffic Manager, the Traffic Manager will add a Record-Route header field to it so that future requests that are part of this session will be routed through the Traffic Manager. All of these requests and the corresponding responses can be manipulated with RuleBuilder and TrafficScript rules.

In a typical SIP session, all the control messages such as INVITE and BYE will pass through the Traffic Manager, but the session data itself will not. This is the default mode of operation for SIP virtual servers in the Traffic Manager.

- **Full Gateway** - Select this option to make all session data pass through the Traffic Manager. This mode is useful if all your SIP clients are on the same network as your Traffic Manager and the Traffic Manager is acting as a gateway to the Internet.

  In this mode, all control messages such as INVITE and BYE will pass through the Traffic Manager. The Traffic Manager will modify any Session Description Protocol information contained in the body data of a request or response so that the session data also passes through the Traffic Manager.

  You can choose to specify a port range that will be used for the session data. You can also specify how long the Traffic Manager will wait before closing the session data connection when no data is being sent over it.

Additional SIP Settings

The connection management page also offers some additional settings that affect the behavior of the Traffic Manager. These settings are:

- **sip_transaction_timeout** - SIP requests and responses are grouped into transactions. There are two types of transactions in SIP: INVITE transactions and Non-INVITE transactions. An INVITE transaction consists of an INVITE request, followed by a series of responses. If the final response was not a '200 OK' response then it is followed by an ACK request. A Non-INVITE transaction consists of a request followed by zero or more responses. A client can send several different transactions over the same connection.

  When no further messages that are part of a particular transaction have been seen for the time specified in **sip_transaction_timeout**, the Traffic Manager will reclaim the resources associated with the transaction.

  If the transaction is not complete when it times out, the Traffic Manager will issue a 408 request timeout response to the client.

- **sip_rewrite_uri** - If set, this option tells the Traffic Manager to replace the URI of incoming requests with that of the back-end node the Traffic Manager has selected. For example, if a request arrives with the URI 'sip:bob@example.com' and the Traffic Manager is responsible for the domain example.com, then the request that is sent to the back-end node will be sip:bob@192.0.2.1:5070 if the back-end node's IP is 192.0.2.1 and its SIP server is running on port 5070.

This setting is useful if your SIP proxies do not check the domain of the user when looking up their location.

- **sip_follow_route** - If set, this option tells the Traffic Manager to follow the routing information contained within incoming requests. If there is a Route header the request will be sent to the corresponding destination. If there is no Route header, the request will be sent to the destination specified by the URI. If the URI points to the Traffic Manager then the request will be sent to the pool as normal.

  This behavior can be overridden by the **sip_dangerous_requests** setting. For example, if a request arrives that contains body data and a Route header, the Route will be ignored and the request sent to the pool if **sip_dangerous_requests** is set to send dangerous requests to a back-end node.

  If **sip_follow_route** is set to No then all requests will be sent to the pool and will have a Route header added that corresponds to the chosen node.

- **sip_dangerous_requests** - SIP requests contain their own routing information, and as such it is possible for them to be used to attack a remote computer through an intermediate machine. The Traffic Manager considers a request dangerous if its next destination is an external IP and the request contains body data.

  The Traffic Manager can handle potentially dangerous requests in several ways:

  1) It can send the request to a back-end node. This option is useful if your SIP proxy servers are responsible for domains that do not resolve to the Traffic Manager's IP. This is the default setting for potentially dangerous requests.

  2) It can send a 403 Forbidden response back to the client. This option is useful if you want to reject any requests that are not addressed to your Traffic Manager's domain.

  3) It can forward the request to its target URI**.** Selecting this option means SIP requests will be routed normally. Use this setting when you want your virtual server to handle outbound SIP traffic.

- **sip_max_connection_mem** - A client can send multiple SIP requests on the same connection and receive responses to them in any order. To stop a client from having an excessive number of active requests awaiting a response, the Traffic Manager can limit the maximum amount of memory each connection can allocate.

  This setting specifies how much memory the Traffic Manager should allow each connection to use before refusing new requests with a '413 Request Entity Too Large' response. If this value is set to 0 then the memory limitations are removed,

however this will leave the Traffic Manager more vulnerable to a Denial of Service attack.

Communicating with UDP-Based SIP Servers

Typically, UDP-based SIP servers respond to requests much like TCP-based SIP servers – by sending the response from the same port on which they received the original request. This behavior is not explicitly required, however, and some SIP servers might respond from a different port or even from a different IP to the one on which it received the corresponding request.

The Traffic Manager caters for this behavior with a configuration setting 'udp_accept_from' on the **Pools > Edit > Connection Management** page (where the pool is used by a UDP service). Here you can select how the Traffic Manager receives responses to UDP requests, from the following options:

- Only the IP address and port to which the request was sent.
- Only the IP address it was sent to but any port.
- A specific set of IP addresses but any port.
- Any IP address and any port.

These settings provide a varying degree of trade-off between security and compatibility. Allowing responses from multiple IP addresses can pose a greater security risk due to the increased possibility of fraudulent responses, yet certain SIP servers may require this functionality to communicate correctly. You should select the option that most closely matches your requirements.

If you choose to only accept responses from a specific set of IP addresses, you will need to enter a CIDR Mask (such as 198.51.0.0/16) in the box provided.

Important: Please note that requests sent to an IPv4 address will expect a response back from an IPv4 address only. Conversely, requests sent to an IPv6 address will allow a response back in either IPv4 or IPv6 form.

An associated config setting is provided for SIP (over UDP) Health Monitors. See the *Built-in Health Monitors* section of CHAPTER 14 for further details.

CHAPTER 12 Session Persistence

This chapter explains how the Traffic Manager is used to provide persistent sessions (also known as sticky sessions) between clients and back-end servers.

What Is Session Persistence?

Many classes of requests from clients can be load-balanced across a pool of back-end servers. Multiple requests from one client can be shared across the back ends with no disruption to service. However, there are certain exceptions, such as server applications which depend upon storing information about a client locally, which may not readily be load-balanced in this way. These include:

- Web mail applications that track a user's login by storing files on disk.
- Shopping carts that store the contents of a client's cart on disk or in memory.
- Programs or protocols with a higher start-up time than normal (such as Java applications or SSL connections).

In these situations, the Traffic Manager can ensure that requests are mapped to the same back-end server for each request, for the duration of the client's session:

When the Traffic Manager receives a new connection, it uses its load balancing logic to choose a node for that connection. The Traffic Manager then records the chosen node in a session persistence map.

When another connection in the same session is received, the Traffic Manager uses the node that was chosen previously.

In this way, all connections in the same 'session' are pinned to the same back-end node. The session persistence class in use defines how a session is identified, and you can refine this decision using several TrafficScript methods.

Session persistence classes can be used to direct all requests in a client session to the same node. This may be necessary for complex applications, where an application session may be maintained over a number of separate connections. Examples of this include Web-based shopping carts, and many complex UDP-based protocols.

Session persistence should only be used when necessary. It effectively bypasses the load-balancing process for all but new sessions, so unless it is necessary to send all requests from the same client to the same back-end node, response times will be better without session persistence.

Configuring Session Persistence

Session persistence is configured via the Catalog. Session Persistence Classes are added to the catalog, and can then be assigned to a pool, or selected in a TrafficScript rule.

When TrafficScript is not being used, Session Persistence is enabled on a *per-pool* basis. You can define multiple pools, and enable session persistence only for those that require it. Note that a back-end server can be in more than one pool.

Several pools can refer to the same session persistence class to share session persistence mappings. For example, a pool containing some HTTP nodes (on port 80) and a pool containing the same nodes running HTTPS services (on port 443) can share the session persistence mappings for users to the services.

When TrafficScript is being used, Session persistence can be overridden with a TrafficScript command. The Traffic Manager will use the Session Persistence Class specified for the pool which handles the request unless this command is used.

Enabling Session Persistence

First, click the **Catalogs** button, then click the **Persistence** tab to open a list of Session Persistence Classes managed by the Traffic Manager. Next, enter a name in the box labeled **Create new Session Persistence class** and click **Create Class**.

The user interface now shows a summary of the Session Persistence Class settings.

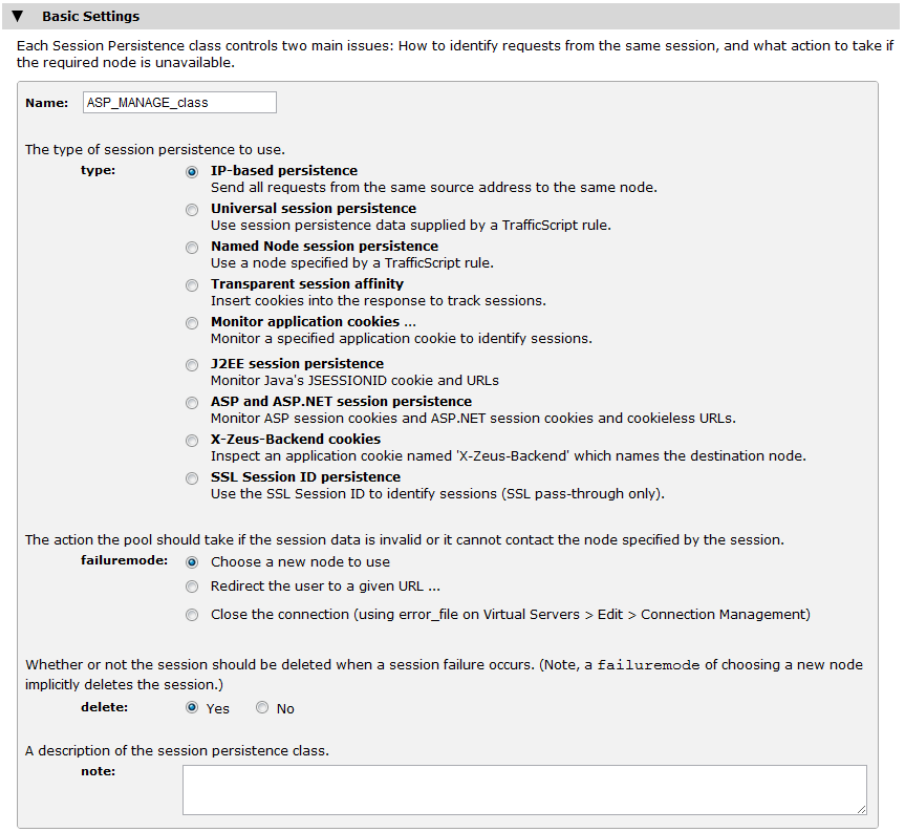

Basic Settings
Each Session Persistence class controls two main issues: How to identify requests from the same session, and what action to take if the required node is unavailable.
Name: ASP_MANAGE_class
The type of session persistence to use.
type:
IP-based persistence
Send all requests from the same source address to the same node.
Universal session persistence
Use session persistence data supplied by a TrafficScript rule.
Named Node session persistence
Use a node specified by a TrafficScript rule.
Transparent session affinity
Insert cookies into the response to track sessions.
Monitor application cookies ...
Monitor a specified application cookie to identify sessions.
J2EE session persistence
Monitor Java's JSESSIONID cookie and URLs
ASP and ASP.NET session persistence
Monitor ASP session cookies and ASP.NET session cookies and cookieless URLs.
X-Zeus-Backend cookies
Inspect an application cookie named 'X-Zeus-Backend' which names the destination node.
SSL Session ID persistence
Use the SSL Session ID to identify sessions (SSL pass-through only).
The action the pool should take if the session data is invalid or it cannot contact the node specified by the session.
failuremode:
Choose a new node to use
Redirect the user to a given URL ...
Close the connection (using error_file on Virtual Servers > Edit > Connection Management)
Whether or not the session should be deleted when a session failure occurs. (Note, a failuremode of choosing a new node implicitly deletes the session.)
delete:
Yes
No
A description of the session persistence class.
note:

Fig 39. Session persistence's basic settings

Selecting a Persistence Method

By default, new Session Persistence Classes are configured with **IP-based Persistence** selected. The following alternatives are available when session persistence is required:

Method	Applicable Protocols	Description
IP-based persistence	All	Send all requests from the same source address to the same node.
Universal session persistence	All	This class allows you to persist sessions based on any information in a request by setting a unique persistence key in a TrafficScript rule.
Named Node persistence	All	With this method, a TrafficScript rule can direct the connection to a specific node.

Method	Applicable Protocols	Description
Transparent session affinity	HTTP, HTTPS	Insert cookies into the response to track sessions.
Monitor application cookies	HTTP, HTTPS	Monitor a specified application cookie to identify sessions.
J2EE	HTTP HTTPS	Monitor Java's JSESSIONID cookie and its URL-rewriting fallback.
ASP and ASP.net	HTTP, HTTPS	This method allows both types of ASP sessions to be processed using session persistence.
X-Zeus-Backend cookies	HTTP, HTTPS	Inspect an application cookie named `X-Zeus-Backend`, which names the destination node.
SSL Session ID persistence	SSL protocols	Use the SSL Session ID to identify sessions.

IP-Based Persistence

When IP-based persistence is selected, the Traffic Manager will track the originating IP address for each request to this pool. If the Traffic Manager has already received traffic from this address, it will map requests to the same back-end server it used previously.

Some specialized mobile clients may change their IP address during a session, and some clients may share IP addresses (for example, via a proxy server), so this may not be suitable for some environments.

However, since IP-based persistence is completely transparent and is not protocol-dependent, it is useful for guaranteeing persistent connections for difficult protocols or applications.

IP session maps are shared by all Traffic Manager machines in a cluster. Requests received by different Traffic Manager machines will be directed to the correct node, and if one Traffic Manager fails, the other Traffic Managers are aware of the IP session maps it was maintaining.

Universal Persistence

Universal session persistence uses a persistence key, defined in a TrafficScript rule, and seeks to direct all connections with the same persistence key to the same back-end node.

The persistence key is specified in a rule by the `connection.setPersistenceKey()` TrafficScript function. This allows persistent sessions based on any information in a request; see the *Universal PHP Persistence* section of CHAPTER 12 for an example.

Universal session persistence maps are shared by all Traffic Manager machines in a cluster. Requests received by different Traffic Manager machines will be directed to the correct node, and if one Traffic Manager fails, the other Traffic Managers are aware of the universal session maps it was maintaining.

Important: Note that when using Universal Session Persistence with the SIP protocol over UDP, session persistence will only be applied to the first request sent by a client. All subsequent requests from the same client will be sent to the same node as the first request, even if they have been assigned a different session persistence key. Universal Session Persistence should therefore be used with SIP only when you want all requests from the same client to be directed to the same server.

Named Node Persistence

Named Node session persistence can be used to direct the connection to a specific node in a pool. It gives very fine-grained control over how requests are routed.

A TrafficScript rule can use the `connection.setPersistenceNode()` function to specify precisely which node should be used in the session persistence decision. For example, when a new connection for a Remote Desktop service is received, a rule could query an external database to determine the node which hosts that user's desktop, and then use the `connection.setPersistenceNode()` function to direct the connection to that node.

Transparent Session Affinity

Transparent session affinity inserts cookies into the HTTP response to track sessions. This is generally the most appropriate method for HTTP and SSL-decrypted HTTPS traffic, because it does not require the nodes to set any cookies in their response.

Transparent session maps are stored in a client-side cookie named **X-Mapping-xxxxxx**, where 'xxxxxx' is an opaque string that identifies the session persistence

class and the value of the cookie is an opaque string that identifies the preferred node. All Traffic Manager machines in a cluster will inspect the value of this session cookie and send the session to the same server node.

Monitor Application Cookies

Application cookie persistence monitors a named cookie in the HTTP response from the node. For example, PHP applications may generate session cookies named "PHPSESSID" that the clustered PHP application could track and index session state. This persistence method directs a request to the same server node if it contains an application cookie. You need to specify the name of the application cookie you wish to monitor.

Application cookie session maps are stored in a client-side cookie **K-CookieName-xxxxxx**, where CookieName is the name of the monitored cookie, 'xxxxxx' is an opaque string that identifies the session persistence class and the value of the cookie is an opaque string that identifies the preferred node. All Traffic Manager machines in a cluster will inspect the value of this session cookie and send the session to the same server node.

If the back-end server changes the value of the application cookie, the session is still valid and clients will continue to be directed to the same server node.

J2EE JSESSIONID Cookies/URL

J2EE JSESSIONID cookies/URL persistence monitors both the JSESSIONID cookie (as for application cookie persistence) and the jsessionid path parameter. These are defined in the Java extension specification (v2.4) and are used by Java extension containers such as BEA WebLogic, IBM WebSphere Application Server, JBoss/Tomcat and others.

J2EE session maps are shared by all Traffic Manager machines in a cluster, so requests received by different Traffic Manager machines will apply the same sessions.

ASP.net Session Persistence

ASP (Active Server Pages) is a server-side scripting protocol created by Microsoft to handle dynamically generated Web pages. In order to use ASP Session management in a server cluster, the *same* Web server must handle all requests coming from a user for the life of the session.

ASP sessions can use cookies or can be cookieless, depending on numerous factors, such as the lack of support of cookies by the browser or the user disabling them voluntarily.

- **For cookie-based sessions:** The Traffic Manager's ASP Session Persistence class detects and uses the cookie to identify the client's session.
- **For cookieless sessions:** The Traffic Manager's ASP Session Persistence class detects and uses the ASP identifier embedded in URLs generated by the ASP application.

ASP session maps are shared by all Traffic Manager machines in a cluster, so requests received by different Traffic Manager machines will apply the same sessions.

Note that if you have several distinct ASP applications hosted behind a single hostname, each application will generate its own cookie. You will need to have a separate ASP.Net session persistence class for each application so that mappings between cookies and nodes are managed independently for each application. For example, you may inspect the request URL to determine which application is being accessed and select the session persistence class using the TrafficScript function `connection.setPersistence()`.

X-Zeus-Backend Cookies

X-Zeus-Backend cookie persistence looks for a cookie named **X-Zeus-Backend** in each application request. If the cookie is present, and contains the name of a node in the current pool, the request is sent to that node. The cookie can be inserted either by a back-end server or by a TrafficScript rule.

Note: This persistence method is deprecated, and provided only for backward compatibility with Zeus Load Balancer persistence cookies.

SSL Session ID Persistence (SSL Pass-Through Only)

The SSL session ID persistence method sends all SSL traffic with the same SSL session ID to the same server node. It is only applicable to SSL pass-through traffic, not SSL-decrypted traffic.

The SSL session ID persistence method reduces the number of SSL handshake operations your nodes perform. SSL handshakes are expensive in terms of CPU time, network bandwidth and latency.

SSL Session ID session maps are shared by all Traffic Manager machines in a cluster. Requests received by different Traffic Manager machines will be directed to the correct node, and if one Traffic Manager fails, the other Traffic Managers are aware of the SSL Session ID session maps it was maintaining.

SSL session ID persistence is not appropriate for application-level session persistence because many SSL clients regularly renegotiate their SSL session ID. To achieve application-level session persistence you should either use IP-based session

persistence, or decrypt the traffic and use universal session persistence or an HTTP method if applicable.

Resolving Session Persistence Maps to Nodes

When the Traffic Manager receives a new connection and there is no session persistence information, the Traffic Manager uses its load balancing logic to choose a node for that connection. The chosen node (IP address and port) is recorded in the session persistence mapping, which is either internal (in the case of IP, Universal or SSL session persistence) or in an external cookie (Transparent, Application cookie or X-Zeus-Backend cookie).

When another connection in that session is received and the Traffic Manager is ready to forward that connection to a node in a pool, the Traffic Manager inspects the session persistence class in use to determine if a particular node should be used.

If a node with the same IP address and port exists in the pool for the connection, then the Traffic Manager sends the connection to that node.

If there is not an exact match, the Traffic Manager searches the pool for any nodes with the same IP address (but different ports). If just one such node exists, the Traffic Manager sends the connection to that node.

This allows session persistence information to be shared between different pools with different nodes types.

For example, a Web-based application may use an HTTP interface to manage items in a shopping cart, and a secure HTTPS interface to manage payment. Session persistence requirements may dictate that users must be directed to the same physical machine for both HTTP and HTTPS traffic.

Both services (HTTP and HTTPS) could reference the same session persistence class. When a user first connects to the HTTP service, the Traffic Manager would use the HTTP pool. A session would be established with a particular node (IP address, port 80) in that pool.

When the user accesses the HTTPS service, the Traffic Manager might use a different pool containing HTTPS nodes. The session persistence class will then direct the request to the same physical machine (IP address, port 443) in the HTTPS pool.

Node Failure Options

Sometimes, the node required by the session persistence mapping may not be available. For example, it may be marked as 'failed' by a monitor, or the Traffic Manager may be unable to connect to it.

The Traffic Manager provides alternative actions for sessions currently using that node. At the bottom of the Session Persistence page are radio buttons to select the action:

Choose a New Node to Use

The pool discards the session map and chooses a new node, using the current load-balancing algorithm.

Close the Connection

The pool immediately closes the connection. HTTP traffic will send an error file of pools.

You can choose whether to discard or remember the session map. If you remember it, then if the client returns and the session's node is available again, the request will be sent to the session's node.

Send an HTTP Redirect (HTTP Only)

The pool sends an HTTP 302 redirect to the configured location (URL) as a response to the request. The resource at the redirect location could display a message, or cause the user to log in again and establish a new session.

You can choose whether to discard or remember the session map. If you remember it, then if the client returns and the session's node is available, the request will be sent to the session's node.

Draining Connections

Sometimes it is necessary to take a back-end server out of service; for example, to upgrade software, perform hardware maintenance or to decommission or repurpose it.

A pool's Connection Draining capability is designed to facilitate this. When you mark a node as 'draining', the Traffic Manager stops sending it any new connections. However, any connections that are in a session previously established to that node are still sent to that node.

This allows you to safely remove a node from a pool without interrupting either ongoing connections, or longer-term established sessions. The **Activity** section of the Admin Server provides reports so that you can discover how long a draining node has been idle, and then make a judgment as to whether all sessions have completed.

For example, suppose you have an e-commerce service and you use session persistence to tie individuals' sessions to particular back-end nodes. You mark one of your nodes as draining.

No new sessions are established with that node.

Existing, established sessions are allowed to continue with that node.

After 60 minutes, you inspect the **Draining Nodes** page in the **Activity** section of the Admin Server. You observe that the node has been idle for 35 minutes. It is probably safe to conclude that all established sessions have now completed, and you can remove the node safely.

In practice, the time periods involved in determining whether a node has finished draining will be very much dependent on the application and user behavior. In setting up the system, an administrator will have to decide these values in relation to the desired use.

The *Draining and Disabling Nodes* section of CHAPTER 5 describes how connection draining is configured in a pool.

Sizing the Session Persistence Caches

Some session persistence methods use client-side cookies to store the session persistence data. The remaining session persistence methods use caches in the Traffic Manager, shared automatically across a cluster, to store session persistence mapping.

The caches are fixed in size. When a cache fills up, the oldest (least recently used) entry is discarded when a new entry is added. Cache sizes are configured in the **System > Global Settings** page, in the **Cache Settings** section:

Section	Setting
IP session persistence	ip_cache_size
Universal session persistence	universal_cache_size
SSL session-id session persistence	ssl_cache_size
J2EE session persistence	j2ee_cache_size
ASP session persistence	asp_cache_size

You can monitor the behavior of the caches using the Activity Monitor. The key values to help you size the cache are:

Setting	Description
Entries	The number of entries in the cache
EntriesMax	The configured maximum size of the cache
Oldest	The time since the least recently used entry was last used

Once the Traffic Manager has processed a number of sessions, it is normal for the cache to completely fill up. The **Oldest** value will indicate the current session expiry time – how long entries are retained without being used before they are discarded.

To help you size your cache, you need to consider the rate at which new entries are added to your cache (entries per second) and the length of time (in seconds) that you want these entries to be retained since they were last used (the session expiry time). Multiply these two values together to get an estimate of the required cache size, and monitor the **Oldest** value to check that entries are not prematurely discarded because the cache has filled.

If you need fine-grained control of session persistence records, the most effective means is to use client-side cookies with specific expiry times, and tie the session persistence to the presence of the cookie.

Using Session Persistence with Multi-Hosted Traffic IP Addresses

If you use multi-hosted Traffic IP Addresses with the 'consider source port' setting enabled, then requests from one source IP address (i.e. from a client) will be handled by all of the Traffic Managers in your cluster (see the *Creating a Traffic IP Group* section of CHAPTER 6).

In this situation, session persistence methods that rely on state sharing (session maps are shared between the Traffic Managers) may not work reliably. There is a short delay before session information is propagated across the cluster, and clients may visit several Traffic Managers during this period, resulting in corruption of the client's session persistence mapping.

Session persistence algorithms that depend on state sharing are:

- IP-based Session Persistence
- Universal Session Persistence
- SSL SessionID Session Persistence
- J2EE and ASP.NET Session Persistence

Do not use the 'consider source port' setting in any multi-hosted Traffic IP addresses that are used by services that use any of the above session persistence methods.

Session Persistence with UDP protocols

UDP protocols are not connection oriented, but you will often desire that UDP datagrams in the same session are routed to the same back-end server.

In the **Virtual Server > Connection Management** settings, the two settings udp_timeout and udp_response_datagrams_expected are used to inform the Traffic Manager what the UDP session should look like.

The UDP session lasts until the number of response datagrams observed is equal to the setting udp_response_datagrams_expected. For example, if a session completes once the server has sent one datagram to the client, set udp_response_datagrams_expected to "1".

If the session is long-lived, set udp_response_datagrams_expected to "-1". The UDP session times out if no further UDP traffic has been observed within udp_timeout seconds.

You can use Session Persistence of various types (typically IP-based or Universal) with UDP if you want to track sessions for longer periods of time.

Examples

Universal PHP Persistence

A TrafficScript rule can inspect incoming HTTP requests, and select a back-end node based on the request.

The example in the *Routing by Content Type* section of CHAPTER 8 gives the TrafficScript rule to select a pool based on filename extension. This rule can be modified to pass session persistence data to a chosen Session Persistence Class.

Create a Session Persistence Classes called `PHP`, and set to use universal session persistence (see the *Configuring Session Persistence* section of CHAPTER 12). Next create a TrafficScript rule in the catalog as follows:

```
$path = http.getPath();
if( string.endsWith( $path, ".php" )) {

  # Persist on the PHPSESSID cookie, IP address and
  # user agent
```

```
  $phpCookie = http.cookie( "PHPSESSID" ) .
               connection.getRemoteIP() .
               http.getHeader( "User-Agent" );

  # Set the persistence key to our unique value
  conection.setPersistenceKey( $phpCookie );

  # Select the PHP Session Persistence Class
  connection.setPersistence( "PHP" );
}
```

Apply this rule to the virtual server handling your traffic. The string passed to the `pool.use()` function (`$phpCookie`) is the session persistence data used by the pool.

Requests for files with other extensions (such as `.html` or `.jpg`) are ignored by the above rule. These requests are passed on to the other rules used by the virtual server, or to its default Session Persistence Class as set in the default pool.

Also note that only one pool is required, and the Session Persistence is managed entirely by the two separate Session Persistence Classes.

CHAPTER 13 SSL Encryption

This chapter explains how to use Secure Sockets Layer (SSL) encryption with the Traffic Manager. It includes a description of SSL and how to use the Traffic Manager to manage authentication and encryption, as well as the storage of certificates.

Overview of SSL

SSL (Secure Sockets Layer) is a protocol used to send traffic securely over the Internet. Traffic is encrypted using a key agreed between the server and client machines.

SSL provides several advantages:

- Server authentication
- Client authentication
- Encrypted data transfer

SSL can be used with almost any TCP/IP protocol, but is most commonly used to secure HTTP (Web) traffic, forming the HTTPS protocol.

Server Authentication

A server identifies itself for SSL communications using an ***SSL certificate***. This certificate contains the name and location of the organization and its DNS name, and gives the client assurance that they are accessing the correct site.

An SSL certificate can be ***self-signed*** by the organization that owns it. However, without independent verification, the certificate will not automatically be trusted by a client. To be trusted, it must be signed by a recognized, independent ***certificate authority (CA)*** such as Verisign or Thawte. The organization sends a ***certificate signing request (CSR)*** to the CA, which carries out thorough checks on the details in the certificate, and may also inspect the organization's financial records. Note that certificate authorities charge for this service.

The client might try to find out whether the server certificate is revoked using the ***Online Certificate Status Protocol (OCSP)***. Using the TLS status_request extension, you can include this check inside the TLS handshake (known as *OCSP Stapling*), so that a separate connection from the client to an OCSP responder is not needed. To enable OCSP stapling, see "Configuring OCSP" below.

What is OCSP?

OCSP is an Internet protocol used for obtaining the current validity of an SSL certificate at the point of use. It was created as an alternative to ***Certificate Revocation Lists (CRLs)*** (see "CRLs in Client Authentication" on page 211) to address some of the inherent shortcomings of that method, such as the limitation that updates must be frequently downloaded to keep the list current.

When users attempt to access a secure service, they send an HTTP request to an OCSP server (known as a *Responder*) for the certificate's status information. This request is packaged in the form of an *ASN.1* message, optionally signed with a certificate, and sent to the responder. In return, the responder sends back a response of "good," "revoked," or "unknown".

You can find further information on OCSP at http://tools.ietf.org/html/rfc2560.

Client Authentication

In some cases, you might want to only allow certain approved people to access your service: for instance, a company intranet or extranet. To achieve this, you can require the client to provide an SSL certificate signed by a trusted certificate authority.

The Traffic Manager uses CRLs and OCSP to ensure the validity of these certificates.

CRLs in Client Authentication

Within the Traffic Manager your trusted certificate authorities are held in a catalog. Each certificate authority can distribute ***certificate revocation lists (CRLs)***, which are also held in this catalog. Certificates usually have a fixed validity period, such as 12 months, but sometimes a certificate is cancelled before it expires. In this case the certificate authority adds it to a certificate revocation list, so that it will no longer be trusted.

OCSP in Client Authentication

You can use OCSP to check the current status of a client certificate. Unless the certificate is reported as being good, the SSL connection is terminated. To configure OCSP for your secure services, see "*Configuring OCSP*" in CHAPTER 13.

Encrypted Data Transfer

Once server and client are satisfied with each other's identity, they agree on an encryption key to use for data transfer. This is different from their identification keys for reasons of efficiency. Data is encrypted before transfer so that a third party cannot read it. In addition, SSL has reliability features that ensure that any disruption to the data stream is detected. These features give client and server confidence that their communication is private, and has not been corrupted.

SSL Features in the Traffic Manager Traffic Manager

Decryption and Encryption

The Traffic Manager can decrypt SSL traffic within the virtual server. This can be useful for two reasons:

- After decryption, the Traffic Manager's traffic analysis features can be used on the whole request. Service protection methods can filter for malicious content, viruses or web worms; and rules can inspect the headers and body of the request to make an informed routing decision. Without decrypting the packets very little information is available.
- Decryption requires processing power. It may be more efficient if the Traffic Manager decrypts requests before passing them on to the nodes, reducing the load on the back-end servers.

If your virtual server is decrypting SSL traffic in order to use rules, you may wish to encrypt it again before sending it to the nodes. This encryption is handled by the pools created on your Traffic Manager, providing complete end-to-end security in your system.

SSL Certificates Catalog

The Traffic Manager provides a catalog store, containing sets of objects your Traffic Managers can use when handling services.

Using catalogs, the Traffic Manager provides a centralized store of SSL server certificates, client certificates, certificate authorities and certificate revocation lists.

SSL Decryption Wizard

The **SSL Decrypt a service** wizard provides a step-by-step process to correctly configure SSL decryption. The wizard performs the following:

- Enables SSL decryption for an SSL virtual server.
- Assigns an SSL certificate from the Catalog to the virtual server.
- Enables SSL encryption for the default pool used by the virtual server.
- Changes all protocol types to the underlying (non-SSL) protocol type.

Note: To use the **SSL Decrypt a service** wizard, you must have at least one SSL virtual server (of any type). If you do not have the necessary SSL certificates, the Traffic Manager helps you create a certificate from the wizard.

To use the **SSL Decrypt a service** wizard:

1. Adjacent to the **Help** link in the toolbar, click the "Wizards" drop-down list and choose **SSL Decrypt a service**. A new window opens which explains that the wizard configures the service to be decrypted on receipt, and re-encrypts traffic before being passed to a pool. Click **Next**.
2. Select the service you want to decrypt. Note that only virtual servers that use an SSL protocol are listed. Click **Next**.
3. Choose which certificate to use to decrypt the incoming requests (or click **Create New** to add a certificate now). Click **Next**.
4. Choose a protocol type. This is the underlying decrypted protocol. For instance, HTTPS requests are decrypted to HTTP internally, so choose the underlying protocol HTTP as this matches the protocol type you are accepting, without the SSL wrapping.
5. Click **Finish** to complete the wizard.

Configuring SSL Certificates

The following configuration options are offered from the **Catalogs > SSL** page on the Admin Server. Note that the following sections refer to functionality applicable to both Client and Server SSL certificates:

- *Server SSL Certificates* are used to identify SSL-encrypted services hosted by the Traffic Manager.
- *Client SSL Certificates* are used when the Traffic Manager needs to authenticate itself against an SSL node.

Creating a New Self-Signed SSL Certificate

To create a new self-signed SSL certificate:

1. Click the **Catalogs** button on the top bar of the Admin UI. Click the **SSL** tab. From here, click the **Edit** button for either **SSL Certificates Catalog** (for Server certificates) or **SSL Client Certificates catalog** depending on your requirements.
2. The Traffic Manager lists any existing certificates that have been configured, and allows you to create or import a certificate. For testing purposes and internal use, an SSL certificate can be ***self-signed***. This means that the certificate has not been signed by a trusted third party and should not be relied upon as a means of authenticating the server.

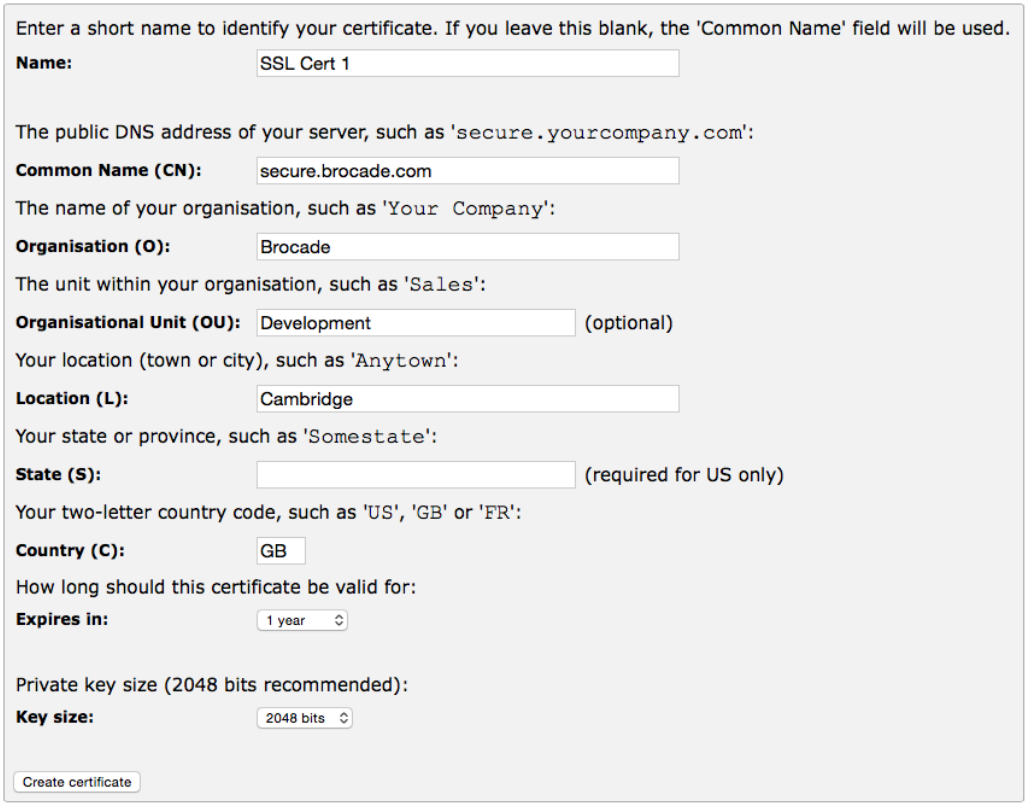

Enter a short name to identify your certificate. If you leave this blank, the 'Common Name' field will be used.
Name:
SSL Cert 1
The public DNS address of your server, such as 'secure.yourcompany.com':
Common Name (CN):
secure.brocade.com
The name of your organisation, such as 'Your Company':
Organisation (O):
Brocade
The unit within your organisation, such as 'Sales':
Organisational Unit (OU):
Development
(optional)
Your location (town or city), such as 'Anytown':
Location (L):
Cambridge
Your state or province, such as 'Somestate':
State (S):
(required for US only)
Your two-letter country code, such as 'US', 'GB' or 'FR':
Country (C):
GB
How long should this certificate be valid for:
Expires in:
1 year
Private key size (2048 bits recommended):
Key size:
2048 bits
Create certificate

Fig 40. Main settings for a self-signed certificate

3. Click **Create Self-Signed Certificate / Certificate Signing Request**, and complete the form.
4. Give the certificate a **name** and a **Common Name** (CN), which should be the DNS name of the server that will use this certificate, such as `secure.yourcompany.com`. Most clients show a warning message if the CN in the certificate and the DNS address of the server do not match.
5. Supply values for your **organization, organizational unit** and **address.** These appear in the certificate.
6. When you enter your country code, use the **two-letter ISO country code**[6]. For example, Great Britain is "GB", and Germany is "DE".
7. Set an **expiry date** for the certificate (default, one year) and **a private key size.** Please note that while 2048-bit keys provide a higher level of security, they require more processing power and are not supported in all clients.
8. Click **Create Certificate**. If there are no errors the Traffic Manager will create an SSL certificate and place this in the SSL Certificates Catalog.

[6] As defined in the ISO-3166 standard.

You can verify that the certificate has been added by clicking the expansion tab on the **Catalogs** page, beside **SSL Certificates Catalog**. Each certificate is listed in a table, and the expiry date of the certificate is also indicated.

Managing Certificate Data

You can change any of the values stored within a self-signed certificate. To do this, click an existing SSL certificate in the appropriate Client or Server SSL Certificates Catalog.

In the **Edit Certificate** box, change any of the existing values for the certificate and then click **Update Certificate**.

Note: You cannot edit a certificate signed by a Certificate Authority, as this would invalidate the signature.

You can also copy an existing SSL certificate, and assign a new certificate name to the copy. To do this, click an existing certificate in the SSL Certificates Catalog. You may need to expand the Client or Server SSL Certificates Catalog first, by clicking on the expansion tab.

In the **Copy Certificate** box, enter a new name for the copy and click **Copy Certificate**. The new SSL certificate is added to the relevant catalog immediately.

Creating a Certificate Signing Request

A Certificate Signing Request (CSR) is a formal request made by an individual to a certificate authority (CA) to obtain a digital identity certificate. Certificate Authorities are entities responsible for issuing certificates for use by other parties.

Once you have created your self-signed certificate, you can then create a Certificate Signing Request based on the information in the self-signed certificate:

1. Go to the **Catalogs > SSL > [Server|Client] Certs** screen.

2. Click to edit the required certificate.

3. In the **Certificate Signing** section, click **Export CSR/Sign certificate**.

The window will now show the text for your certificate request:

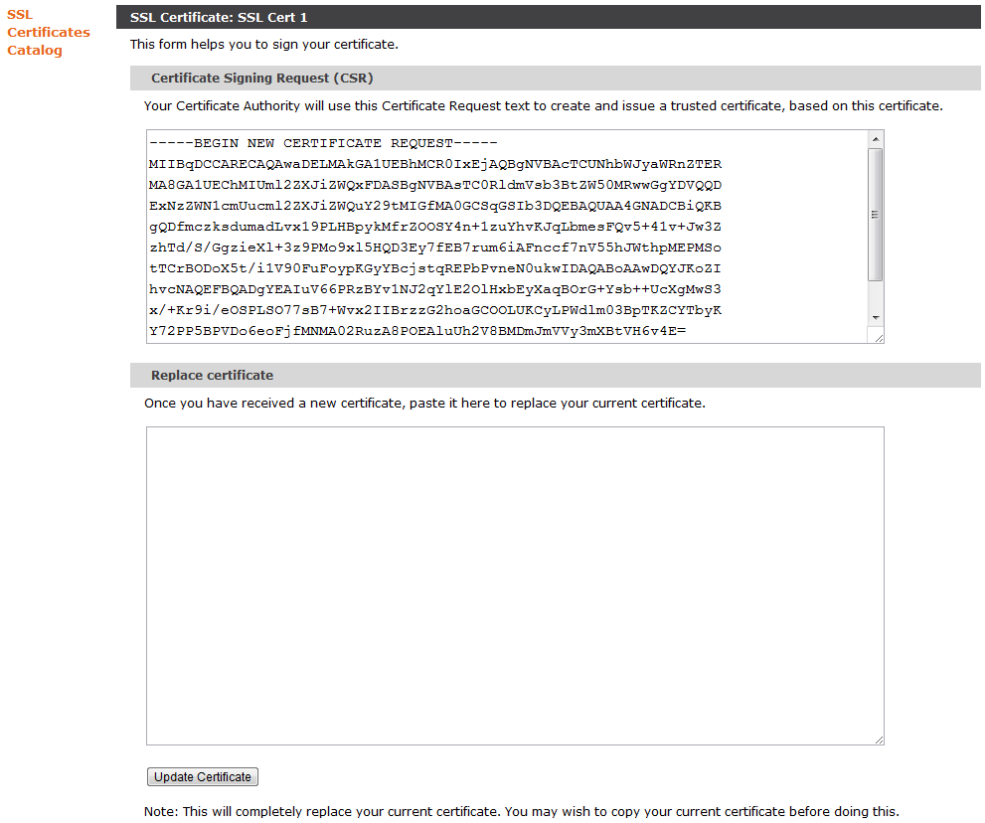

SSL Certificates Catalog
SSL Certificate: SSL Cert 1
This form helps you to sign your certificate.
Certificate Signing Request (CSR)
Your Certificate Authority will use this Certificate Request text to create and issue a trusted certificate, based on this certificate.
-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBqDCCARECAQAwaDELMAkGA1UEBhMCR0IxEjAQBgNVBAcTCUNhbWJyaWRnZTER
MA8GA1UEChMIUml2ZXJiZWQxFDASBgNVBAsTC0RldmVsb3BtZW50MRwwGgYDVQQD
ExNzZWN1cmUucml2ZXJiZWQuY29tMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDfmczksdumadLvx19PLHBpykMfrZOOSY4n+1zuYhvKJqLbmesFQv5+41v+Jw3z
zhTd/S/GgzieXl+3z9PMo9xl5HQD3Ey7fEB7rum6iAFnccf7nV55hJWthpMEPMSo
tTCrBODoX5t/i1V90FuFoypKGyYBcjstqREPbPvneN0ukwIDAQABoAAwDQYJKoZI
hvcNAQEFBQADgYEAIuV66PRzBYv1NJ2qYlE2OlHxbEyXaqBOrG+Ysb++UcXgMwS3
x/+Kr9i/eOSPLSO77sB7+Wvx2IIBrzzG2hoaGCOOLUKCyLPWdlm03BpTKZCYTbyK
Y72PP5BPVDo6eoFjfMNMA02RuzA8POEAluUh2V8BMDmJmVVy3mXBtVH6v4E=
Replace certificate
Once you have received a new certificate, paste it here to replace your current certificate.
Update Certificate
Note: This will completely replace your current certificate. You may wish to copy your current certificate before doing this.

Fig 41. Your certificate request to the Certificate Authority (CA)

The text in the upper part of the window contains an encoding of the information in the stored certificate suitable for the CA to sign. You can copy and paste this text into the request. You may be asked to attach other credentials or extra information to your request.

The Certificate Authority returns a replacement certificate that they have digitally signed with one of their public certificates (or a certificate in a public certificate chain). Paste the textual contents of the returned certificate in the lower part of the window, and click **Update certificate** to complete the process.

Importing a New SSL Certificate

To import an existing certificate, you will require:

- The certificate file.
- The private key file.

These files are in a standard, ***PEM-encoded*** format that can be cut and pasted into a text file for uploading. PEM-encoded certificates are formatted as follows:

```
-----BEGIN CERTIFICATE-----
MIIBPLMAkGA1UEBhMCWkExFTATBgNVBAgTDFdlc3Rlcm4gQ2Fw
ZTESMwZSBUb3duMRQwEgYDVQQKEwtPcHBvcnR1bml0aTEYMBYG
A1UEClcnZpY2VzMRowGAYDVQQDExF3d3cuZm9yd2FyZC5jby56
YTBaMBAQUAA0kAMEYCQQDT5oxxeBWu5WLHD/G4BJ+PobiC9d7S
6pDvAtXdm2j190D1kgDoSp5ZyGSgwJh2V7diuuPlHDAgEDoAAw
DQYJKDQQBf8ZHIu4H8ik2vZQngXh8v+iGnAXD1AvUjuDPCWzFu
pReiq7UR8Z0wiJBeaqiuvTDnTFMz6oCq6htdH7/tvKhh==
-----END CERTIFICATE-----
```

Click **Import Certificate** in the relevant **Client/Server SSL Certificate Catalog** to perform the upload. Provide a name for the imported certificate, and the names of the *Certificate* and *Private Key* files. Now click the **Import Certificate** button to complete the process. The Traffic Manager will add the certificate to the appropriate catalog and automatically distribute the key data securely to all Traffic Managers.

Important: Do not forget to delete temporary copies of your SSL private key that you may have created during the import process. The private key is valuable and should never be stored in an insecure location.

When a certificate signed by a Certificate Authority is not signed directly by the **root certificate** it may be necessary to send clients both the signed certificate and the Certificate Authority's intermediate certificate.

Working with Intermediate Certificates

Web Browsers and other SSL clients are preconfigured with a set of **root certificates** from Certificate Authorities that they trust. They will allow the user to connect to an SSL service that uses an SSL Server Certificate signed by one of the trusted Certificate Authorities. Similarly, back-end SSL nodes may be pre-configured to authenticate Client SSL certificates presented by the Traffic Manager.

However, for ease of management and improved security, many certificate authorities use **certificate chains**. The SSL certificates they distribute are not signed directly by a root certificate; rather they are signed by an intermediate certificate which is itself signed by the root. This forms a chain of trust from the SSL certificate back to the trusted root certificate. In some cases, a chain of two or more intermediate certificates is used between the SSL certificate and the root.

Web clients and SSL nodes are generally not equipped with the intermediate certificates. If, for example, a Web client is presented with a server certificate, it has no way of verifying that it was signed (albeit indirectly) by a trusted root certificate. In this case, the SSL server (i.e. the Traffic Manager) must present the entire or partial SSL certificate chain so that the client can verify the SSL certificate – the certificate

chain includes the SSL certificate and all intermediate certificates. It is not necessary to include the root certificate.

Your certificate authority will inform you if a chain of certificates is used. You need to upload all of the intermediate certificates to the relevant Server or Client SSL certificate catalog:

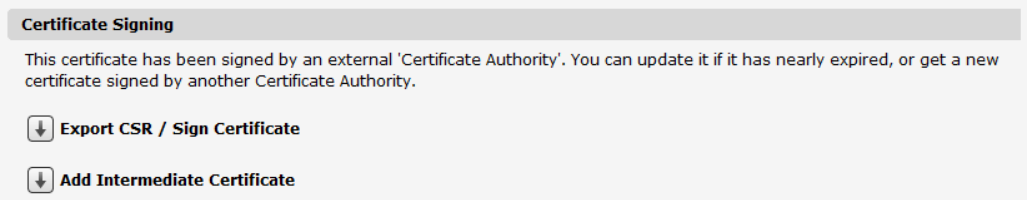

Certificate Signing
This certificate has been signed by an external 'Certificate Authority'. You can update it if it has nearly expired, or get a new certificate signed by another Certificate Authority.
Export CSR / Sign Certificate
Add Intermediate Certificate

Fig 42. Adding intermediate certificates

The Traffic Manager will verify that the intermediate certificate you upload has signed the current SSL certificate (or the previous intermediate certificate in the chain).

Managing Certificate Authority Certificates and CRL Files

The Traffic Manager can request that remote clients who connect over SSL provide a client certificate to authenticate themselves. You will need to configure the Traffic Manager with the public certificates from the Certificate Authorities you trust, and optionally any Certificate Revocation Lists (CRLs) that they distribute.

From the **Catalogs** page, click the **Edit** button for the **Certificate Authorities and Certificate Revocation Lists Catalog**.

The Traffic Manager can import the CA or CRL file using one of three methods:

- A file uploaded via a web browser to the Traffic Manager
- A URL for a file which the Traffic Manager will download directly
- A file manually pasted into the **File Contents** box (PEM-encoded)

Select one of the methods and click **Import File**. The Traffic Manager will then import the CA or CRL file and propagate the new information to all Traffic Managers.

SSL Decryption

The Traffic Manager can decrypt and re-encrypt SSL traffic dynamically as it proxies requests between clients and back-end pools; it uses certificates stored in the catalogs, including server and client certificates. After decrypting traffic, perhaps to apply TrafficScript rules, the Traffic Manager can re-encrypt the data if required before passing it to the nodes. This allows extremely flexible management of requests without compromising security requirements on potentially untrusted networks. The Traffic Manager thus provides a fully transparent SSL gateway.

Setting Up SSL Decryption

SSL decryption is performed by a virtual server. When enabled, SSL decryption will use additional CPU resources in the Traffic Managers that perform the decryption.

To enable SSL decryption, click **Services** on the top bar of the Admin Server. Click the **Virtual Servers** tab to open a list of all virtual servers the Traffic Manager manages. Next, click the **Edit** button for a virtual server that supports SSL.

In the list of configuration options for the virtual server, click the **Edit** button for **SSL Decryption**.

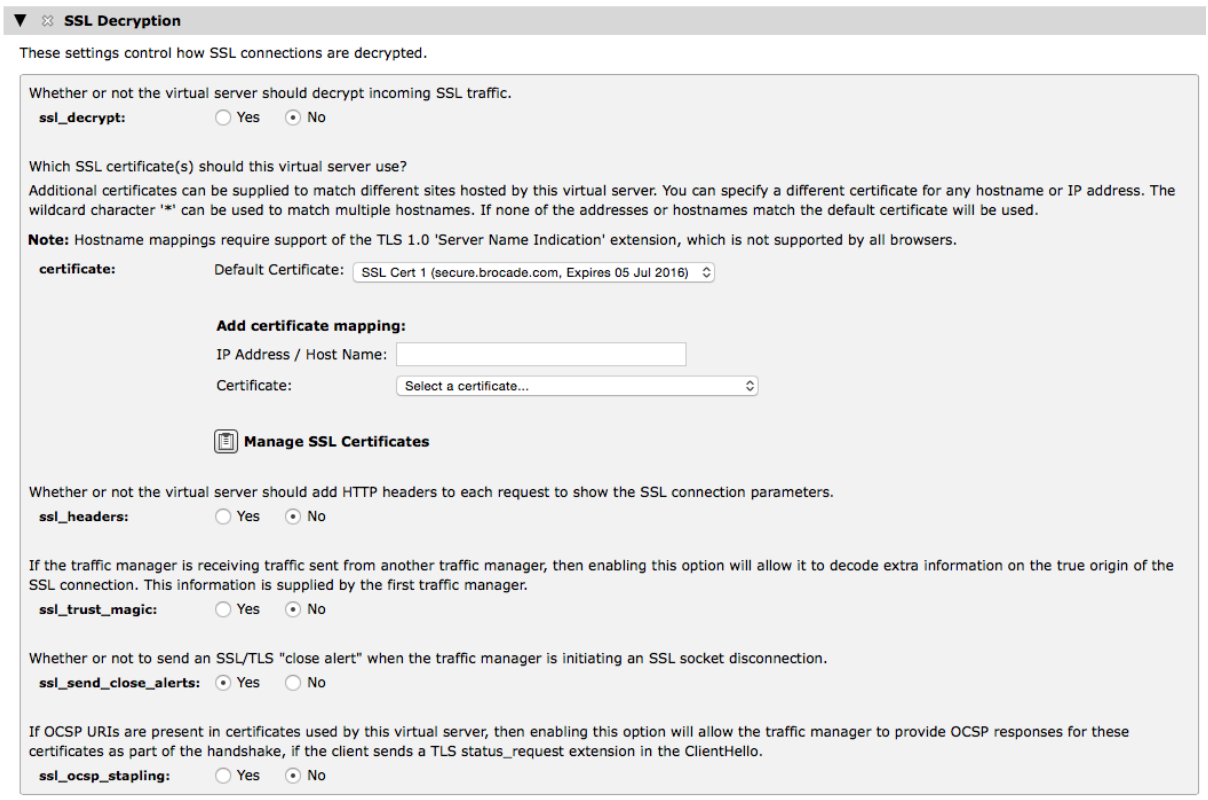

SSL Decryption
These settings control how SSL connections are decrypted.
Whether or not the virtual server should decrypt incoming SSL traffic.
ssl_decrypt: Yes No
Which SSL certificate(s) should this virtual server use?
Additional certificates can be supplied to match different sites hosted by this virtual server. You can specify a different certificate for any hostname or IP address. The wildcard character '*' can be used to match multiple hostnames. If none of the addresses or hostnames match the default certificate will be used.
Note: Hostname mappings require support of the TLS 1.0 'Server Name Indication' extension, which is not supported by all browsers.
certificate: Default Certificate: SSL Cert 1 (secure.brocade.com, Expires 05 Jul 2016)
Add certificate mapping:
IP Address / Host Name:
Certificate: Select a certificate...
Manage SSL Certificates
Whether or not the virtual server should add HTTP headers to each request to show the SSL connection parameters.
ssl_headers: Yes No
If the traffic manager is receiving traffic sent from another traffic manager, then enabling this option will allow it to decode extra information on the true origin of the SSL connection. This information is supplied by the first traffic manager.
ssl_trust_magic: Yes No
Whether or not to send an SSL/TLS "close alert" when the traffic manager is initiating an SSL socket disconnection.
ssl_send_close_alerts: Yes No
If OCSP URIs are present in certificates used by this virtual server, then enabling this option will allow the traffic manager to provide OCSP responses for these certificates as part of the handshake, if the client sends a TLS status_request extension in the ClientHello.
ssl_ocsp_stapling: Yes No

Fig 43. Setting up a client's SSL decryption and the use of certificates

The Traffic Manager now displays a list of configuration options for SSL decryption for this virtual server.

Set the virtual server to decrypt SSL traffic using the **ssl_decrypt** radio button.

Select the certificate you wish to use:

- **ssl_headers** - The Traffic Manager includes the option to add HTTP headers with details about the decryption performed (HTTPS decryption only). These describe the client cipher, session ID and whether a valid client certificate was provided, and can be inspected by the back-end application to determine what encryption parameters were used.

Header name	Sample Value
SSLClientCipher	`SSL_RSA_WITH_AES_128_CBC_SHA256, version=TLSv1.2, bits=128`
SSLClientCertStatus	`NoClientCert` or `OK`
SSLSessionID	`8723367551317ABE5443278C6E...` (64 hex characters)

- **ssl_send_close_alerts** - To avoid truncation attacks, the SSL specifications require that the client and server share the knowledge that a connection is ending by sending a close alert. This setting is enabled by default as some clients, especially FTP clients, are strict with close alerts and fail to function properly if the alerts are omitted. Note that close alerts can break some older SSL implementations, including some versions of Microsoft Internet Explorer. Brocade recommends disabling this setting if your services are affected. You can also modify this behavior dynamically by using TrafficScript rules.
- **ssl_ocsp_stapling** - Enable this option to allow the virtual server to include an OCSP certificate status in the TLS handshake, provided that a status_request extension is sent by the client. It also triggers OCSP requests for each of the virtual server's certificates to be made in the background to the responders named in each certificate's *Authority Information Access* extension's OCSP URI. This option is disabled by default.

Click **Update** to apply the changes to the virtual server.

Using Multiple SSL Certificates

The SSL certificate contains a value called the Common Name (CN), and a client will generally check that the common name value matches the DNS name it is using to connect to the service. It will warn the end user if they do not match.

If your virtual server is managing traffic for more than one service (e.g. secure.site1.com, secure.site2.com, etc.), you will want to configure it with multiple certificates and ensure that it sends the correct certificate for the service the client is trying to access.

There are two ways that the Traffic Manager can distinguish between different services:

- **Destination IP address** - secure.site1.com and secure.site2.com could resolve to different IP addresses. The Traffic Manager can listen on both IP addresses and select the certificate to use based on the IP address the connection was received on.
- **TLS Server Name Indication**[7] - TLS (an updated version of SSL version 3) has an optional capability where a client can provide the name of the service it is trying to contact (in plaintext) at the very beginning of the TLS handshake. The Traffic Manager can inspect this name and chose the certificate to use.

Important: The TLS Server Name Indication does not require that each SSL service runs on a dedicated IP address, so it is more scalable and easier to manage. However, at the time of writing, not all common web clients support this feature; clients such as Internet Explorer 7 (Windows XP) will receive the wrong certificate if they attempt to access a service that requires TLS server name support.

Configuring Multiple Certificates

The Traffic Manager will use the primary certificate (configured with the **certificate** setting in the virtual server) by default. You can override this decision by configuring mappings in the **ssl_sites** setting of the virtual server configuration.

You can configure mappings from IP address to certificate; if the client connects to the configured IP address, it will be given the nominated certificate rather than the default one.

You can configure mappings from domain name (or wildcard domain) to certificate; if the client provides a server name in their SSL handshake that matches the domain name, they will be given the nominated certificate.

When the Traffic Manager receives a request it will check the certificate settings to determine the certificate to send. It will chose the first matching certificate: exact matches of server name come first, then wildcard matches of server name, then exact matches of IP address and finally wildcard matches of IP address. If there are no matches, then the default certificate is used.

[7] http://en.wikipedia.org/wiki/Server_Name_Indication

Configuring Ciphers and TLS Versions

For each SSL decrypting virtual server, you can use the **ssl_support_<version>** and **ssl_ciphers** configuration options to configure individually the SSL/TLS versions and the list of ciphers available for secure communication.

Specify your ciphers (in order of preference) in a space-, comma-, or colon-separated list, as shown in the following example:

```
SSL_DHE_RSA_WITH_AES_128_CBC_SHA,SSL_DHE_RSA_WITH_AES_25
6_CBC_SHA,SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
```

With TLS 1.2 enabled, you can also configure a list of *signature algorithms* a virtual server or pool permits when negotiating TLS 1.2 connections. Set your list of algorithms in **ssl_signature_algorithms** on the virtual server "SSL Decryption" page, or in **ssl_signature_algorithms** on the pool "SSL Encryption" page. See the online help for the list of available algorithms.

For virtual servers with **request_client_cert** set, only client certificates signed using one of these signature algorithms are accepted. For pools, the Traffic Manager sends your list of signature algorithms to the node when negotiating a TLS 1.2 connection.

Leave these items set to their default value of "Use the global setting", or *blank* in the case of **ssl_ciphers** and **ssl_signature_algorithms**, to force the Traffic Manager to instead use the global values set in **System > Global Settings > SSL Configuration**.

Additionally, with TLS 1.0 or higher enabled, the Traffic Manager supports the use of Ephemeral Elliptic Curve Diffie Hellman (ECDHE) key agreement. The Traffic Manager uses ECDHE to agree a shared secret in a TLS handshake, similarly to DHE, with the key advantage that equivalent security can be achieved with smaller (and faster to compute) keys.

Ciphers that use Diffie Hellman key exchange (DHE or ECDHE) are only used if the client supports one or more of your signature algorithms.

To specify a preference list of elliptic curves, type a space, comma, or colon separated list into **ssl!elliptic_curves**. The order of the supplied list determines the priority of the curves, and the Traffic Manager selects the first curve from the list that is supported by the client. The available curves are:

- P256
- P384
- P521

If you leave **ssl!elliptic_curves** blank, the Traffic Manager uses a default list of "P256, P384, P521".

This list can be overridden on an individual virtual server or pool level. Use the **ssl_elliptic_curves** setting on the virtual server "SSL Decryption" page, or **ssl_elliptic_curves** on the pool "SSL Encryption" page.

The Traffic Manager can also be configured with a separate elliptic curve preference list for use with internal and admin server communications. To configure this list, use **admin!ssl_elliptic_curves** in **System > Security**. If no list is entered, the Traffic Manager uses **ssl!elliptic_curves** instead.

Client Certificates

SSL (server) certificates are used to identify servers in an SSL transaction; SSL also allows for the use of **Client Certificates** so that clients can identify themselves when required.

By default, the Traffic Manager will not request a client certificate from clients that connect to a virtual server. If required, the virtual server can be configured to request that clients provide client certificates.

- **request_client_cert** - You can specify whether the virtual server should request an identifying certificate from each client, and whether supplying a certificate should be optional or compulsory (i.e. all clients must present a valid client certificate). Note that if this is set to *not request* a certificate (the default setting), a client will be prevented from sending a certificate whether it wishes to or not.
- **client_cas** - When client certificates are in use, the Traffic Manager must be able to verify that the certificate it is given is legitimate. It will inform the client which CAs it trusts and the client will send a certificate signed by one of these CAs (if available). Select the CAs as required.

If the client supplies a certificate which is not recognized by the CAs you specify, the connection is immediately closed and an SSL error is sent to the client.

- **ssl_client_cert_headers** - Once the request has been decrypted, the Traffic Manager can optionally set HTTP headers that record the certificate fields from the SSL Certificate used, and if required additional certificate text from the SSL Certificate.

Header Name	Sample Value
SSLClientCertVersion	3
SSLClientCertSerialNumber	ED41A8

Header Name	Sample Value
SSLClientCertIssuer	C=US, ST=CA, L=San Jose, O=MyOrg
SSLClientCertSubject	C=US, ST=CA, L=San Jose, O=MyOrg
SSLClientCertNotValidBefore	1155304575 (unixtime)
SSLClientCertNotValidAfter	1186840575 (unixtime)
SSLClientCertSubjectPublicKey	RSA (2048 bit)
SSLClientCertSignatureAlgorithm	md5withRSAEncryption
SSLClientCertHash	873FECCB50E0C1D34711ABE65BA70BA7

Finally, if you select the option to include the certificate text as well, the Traffic Manager will place the entire PEM encoded certificate in a header named `SSLClientCert`.

Configuring OCSP

The Online Certificate Status Protocol (OCSP) can be used to check the revocation status of certificates from a centralized server called an *OCSP responder*. It is commonly used as an alternative to Certificate Revocation Lists (CRLs). See the *Client Authentication* section of CHAPTER 13 for further details about OCSP.

The Traffic Manager provides a dedicated section under **Virtual Servers > SSL Decryption** for configuring OCSP. The following general settings are available:

Setting	Description
ssl_use_ocsp	If set to yes, this virtual server will use OCSP to verify that client certificates have not been revoked.
ssl_ocsp_timeout	When contacting an OCSP responder, the Traffic Manager waits for a response for the amount of time specified by this setting. If the OCSP responder has not replied within this time limit, the client's certificate will be rejected.

Setting	Description
ssl_ocsp_max_response_age	If an OCSP response's '`thisUpdate`' field is older than the number of seconds specified by this setting, the response will be considered invalid. If set to `0` (zero), the '`nextUpdate`' field of the response will be used to determine whether it has expired or not. If the time specified in the '`nextUpdate`' field of the response has passed, the response is considered invalid regardless of this setting.
ssl_ocsp_time_tolerance	If the OCSP responder and Traffic Manager's clocks differ slightly then the times specified in the '`thisUpdate`' and '`nextUpdate`' fields of a response might cause the Traffic Manager to consider it invalid. If the responder's clock is a few seconds faster than the Traffic Manager's clock, for example, the '`thisUpdate`' field of a response might appear to the Traffic Manager to be in the future. This setting allows you to specify the number of seconds to permit for clock differences.

The following OCSP responder settings are provided on a per-issuer basis. These allow you to specify different settings for certificates signed by particular issuers in the Certificate Authorities catalog.

Select a certificate authority in the **Add Issuer Specific Settings** section to add settings for that issuer. If the Traffic Manager receives a client certificate with an issuer that is not configured explicitly, the **Default Settings** will be used.

Setting	Description

Setting	Description
OCSP check	This setting is used to enable and disable OCSP checks for the issuer. Selecting 'Always' will cause remote connections to be dropped when an appropriate OCSP responder URL cannot be determined.
Use AIA for URL	Issuers can embed an OCSP responder URL into their client certificates using the X.509 *Authority Information Access* extension. If this setting is enabled, and the extension is available, the Traffic Manager will use it. Otherwise it will use the **Fallback URL**.
Fallback URL	If the responder URL cannot be determined by other means, this URL is used.
Request signing certificate	OCSP requests can be signed so the responder can identify and authenticate the source of the request. You can select a certificate uploaded to the SSL Certificates catalog to sign the request, choose to use the certificate configured under Default Settings, or disable request signing.
Use nonce extension	A unique sequence of data can be added to OCSP requests and responses to ensure an attacker cannot re-send a 'good' OCSP response after a client certificate is revoked. When using the nonce extension you can require the responder to always return a nonce by using 'strict' mode. Otherwise the Traffic Manager will accept the response if it does not contain the nonce. **Important:** Using this extension stops the responder from pre-generating responses to improve performance. Additionally, not all OCSP responders support this extension. If this is the case, a responder may return an 'unauthorized' response.

Setting	Description
Responder certificate	OCSP responses are signed by the responder to prove they are genuine. You can select a specific certificate that responses should be signed by, from those listed in the **SSL Certificate Authorities** catalog. You can instead configure the system to expect the response to be signed by the issuer's certificate by selecting "**Issuing Certificate**". Alternatively, you can select "**Signed by issuing certificate**". This means the certificate must either be the issuer's certificate itself, or be signed by it and have the `id-kp-OCSPSigning` marker in its **extendedKeyUsage** extension, and also have the **id-pkix-ocsp-nocheck** extension.

SSL Session ID Cache

Once an SSL handshake has taken place, the SSL client (e.g. the Web browser) and the SSL server (e.g. the Traffic Manager) record the encryption parameters using an SSL Session ID.

If the SSL TCP connection is terminated and the client reconnects, it may present its SSL Session ID. If the server recognizes the SSL Session ID, it can skip the computationally expensive SSL handshake and resume the SSL session using the cached encryption parameters.

The Traffic Manager manages a cache of SSL session IDs. This cache is configured on the **System > Global Settings > SSL Configuration** section; you can configure the size of the cache (**ssl!cache!size**) and the expiry time (**ssl!cache!expiry**).

You can monitor the behavior of the SSL Session ID Cache using the **Activity > Current Activity** page. Select the **Cache values > SSL Session ID Cache** values to plot on the current activity chart, or monitor these values using SNMP. The **Oldest** value will indicate the time since the least recently used entry in the cache was used. If the cache is full (comparing **Entries** with **EntriesMax**) and the **Oldest** value is

significantly lower than the configured expiry time, then it is likely that your cache is too small and entries are being expired too early.

OCSP Stapling Cache

The Traffic Manager maintains a cache of OCSP responses for server certificates used by virtual servers that have `ssl_ocsp_stapling` enabled.

In the **System > Global Settings > SSL Configuration** section of the Admin UI, you can configure the Traffic Manager's OCSP response caching behavior:

- Set the size of the cache with `ssl!ocsp_stapling!mem_size`.
- Use `ssl!ocsp_stapling!default_refesh_interval` to control how long the Traffic Manager waits before making a new OCSP request, if no response with a "nextUpdate" time in the future is available in the cache.
- Use `ssl!ocsp_stapling!maximum_refresh_interval` to define the maximum interval that can elapse between OCSP requests. After a valid OCSP response is received, the next update is made this many seconds in the future, unless the "nextUpdate" time in the OCSP response is sooner.
- The Traffic Manager provides a signature verification option `ssl!ocsp_stapling!verify_response`. Set this to **Yes** to force the Traffic Manager to verify the signatures and validity periods of OCSP responses before using them for stapling. If verification fails, the Traffic Manager never uses the response for OCSP stapling.
- An OCSP response has a validity period defined by two fields in the OCSP response: "thisUpdate" and "nextUpdate". When `ssl!ocsp_stapling!verify_response` is enabled, responses outside this period are not cached and therefore not stapled. In some environments, clock skew might be a concern. This can cause the local current time to appear to be earlier than the "thisUpdate" field in the OCSP response, causing an incorrect validity failure. Use the `ssl!ocsp_stapling!time_tolerance` setting to define an allowed time margin (in seconds) outside this interval when the Traffic Manager should still consider responses to be valid.

SSL Encryption

A pool can perform SSL encryption before it sends traffic to a back-end SSL server. When enabled, SSL encryption can use additional CPU resources on the Traffic Managers that perform the encryption.

To enable SSL encryption for a pool, go to **Pools > Edit > SSL Settings**.

You can configure the following options:

- **ssl_encrypt**: Set to **Yes** to enable SSL encryption to the back-end nodes.
- **ssl_send_close_alerts**: Set to **Yes** to send an SSL/TLS "close alert" when initiating a socket disconnection.
- **ssl_support_<version>**: Set to **Enabled** to allow this pool to use the designated version of SSL or TLS for connections to back-end nodes. Choose **Use the global setting** to force the Traffic Manager to use instead the equivalent global value set in **System > Global Settings > SSL Configuration**.
- **ssl_ciphers**: Specify the list of ciphers available for secure communications to back-end nodes. Use a space-, comma-, or colon-separated list (in order of preference). Leave the list blank to force the Traffic Manager to use instead the equivalent global value set in **System > Global Settings > SSL Configuration**.
- **ssl_client_auth**: If the back-end server requires client certificate authentication, enable this setting to configure the Traffic Manager to select an appropriate certificate from its local Client Certificates catalog. The server sends to the client a list of Certificate Authorities that it trusts, and the Traffic Manager chooses the first client certificate it finds that is signed by one of the server's trusted CAs; however there is no explicit control over which certificate is used.
- **ssl_server_name**: Set to **Yes** to cause the Traffic Manager to try the TLS 1.0 "server_name" extension, which can help the back-end node to provide the correct certificate. If you enable this setting, the Traffic Manager is forced to use at least TLS 1.0.
- **ssl_strict_verify**: As a protection against man-in-the-middle attacks and server spoofing, the Traffic Manager can validate the SSL certificates used by back-end servers against the CA's in the catalog. Set to **Yes** to instruct the Traffic Manager to reject connections to servers if their certificates have expired, or if the certificate's Common Name does not match the node's IP address, hostname, or one of the names listed in "ssl_common_name_match", or if a CA in the catalog has not signed it.
- **ssl_common_name_match**: A list of hostnames or IP addresses that the Traffic Manager accepts as Common Name in the certificate presented by the server.

Click **Update** to save your changes.

Preserving IP Addresses with SSL Forwarding

When the Traffic Manager forwards HTTP requests, it can optionally insert a header named `X-Cluster-Client-Ip` into the request so that downstream servers can

determine the correct source IP address of the request. If the Traffic Manager decrypts an HTTPS request, it can also insert the `X-Cluster-Client-Ip` header into the request, even if it then re-encrypts the request.

However, if the Traffic Manager forwards an SSL request without decrypting and re-encrypting it, it cannot modify the data inside. This configuration is used with a loopback virtual server, whereby the connections are load-balanced across a cluster of Traffic Manager systems for decryption. In this case, you may use the **ssl_enhance** setting in the pool to add a proprietary header to the SSL connection that contains key connection data that is not preserved, i.e. source IP and port and destination IP and port.

The Traffic Manager system that receives the 'enhanced' SSL connection must be configured with the **ssl_trust_magic** setting in the SSL decryption settings of the virtual servers. This setting will cause the Traffic Manager to strip out the proprietary header, and recognize the correct connection data – source IP and port, and destination IP and port.

The **ssl_enhance** setting can also be used when forwarding SSL connections to Zeus Web Servers, configured with the **ssl_trust_magic** setting.

Use of SSL Cryptographic Devices

The Traffic Manager software variant supports SSL hardware based on the RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki), such as the Thales e-Security nShield Connect.

The Traffic Manager virtual appliance supports the Thales e-Security nShield Connect and the nCipher netHSM network attached HSMs.

All Traffic Manager variants support the Microsoft Azure Key Vault service. Azure Key Vault is offered as a cloud based alternative to traditional hardware security modules. For Azure Key Vaults, observe the following conditions:

- The Traffic Manager supports using keys that are created by the Traffic Manager itself, or valid existing RSA keys migrated into the Azure Key Vault.
- The Traffic Manager supports "RSA-HSM" type keys. These keys are stored on secure hardware at the Azure Key Vault service.
- To create and use RSA-HSM keys, you must use a "Premium" service tier Azure Key Vault. For information and pricing for the available service tiers, see http://azure.microsoft.com/.

- In common with other secure hardware support, the Traffic Manager can use keys stored in an Azure Key Vault for SSL decryption (virtual servers) or SSL encryption (pools). The Traffic Manager cannot use stored keys for other purposes such as SSH or DNSSEC.
- If the Traffic Manager is configured to use FIPS mode (see CHAPTER 31, "FIPS Validation in "), communications with the Azure Key Vault REST API operate in FIPS mode as well.
- To use the Azure Key Vault, the Traffic Manager's DNS must be able to resolve the hostnames of both the Azure Key Vault server and Microsoft's login server. The Traffic Manager must also be able to establish outgoing TCP connections on port 443. If your Traffic Manager system is firewall restricted, you must ensure your firewall is configured to allow this communication channel.

Note: For instructions on setting up and managing an Azure Key Vault, see http://azure.microsoft.com/en-us/documentation/services/key-vault/.

Using an HSM device or Azure Key Vault service offloads SSL computation (the RSA private key decryption) from the Traffic Manager system's CPU onto the SSL cryptographic device to which you connect. Some PKCS#11 devices also provide hardware key management, so that the private key is stored securely on the hardware device and cannot be accessed directly without the correct authentication.

As RSA cryptographic operations being performed on an SSL cryptographic device or service are outside of the Traffic Manager FIPS 140-2 Cryptographic Boundary, you should independently ensure that your cryptographic device or service is sufficiently conformant to FIPS 140-2 for your requirements.

Although the majority of SSL devices require explicit configuration, as described later, the Traffic Manager automatically detects and uses the SSL hardware support present in UltraSPARC T2 processors.

Note: Some Traffic Manager variants, such as the virtual appliance, display a limited set of the following options based on the hardware and services supported for these variants.

Note: Use of SSL devices might not improve the overall performance of your Traffic Manager system. Although such devices offload the RSA calculation from the main CPU cores, the overhead in communicating with the device is not negligible. The Traffic Manager is able to perform many thousands or tens-of-thousands of SSL calculations on general purposes CPUs. The primary benefit of many SSL devices is their ability to store the private key securely.

Configuring the Traffic Manager to Use an SSL Device

To configure the Traffic Manager to use an SSL device or service, click **System > Global settings** and then click **SSL Hardware Support**. Select the desired SSL device in **ssld!library**.

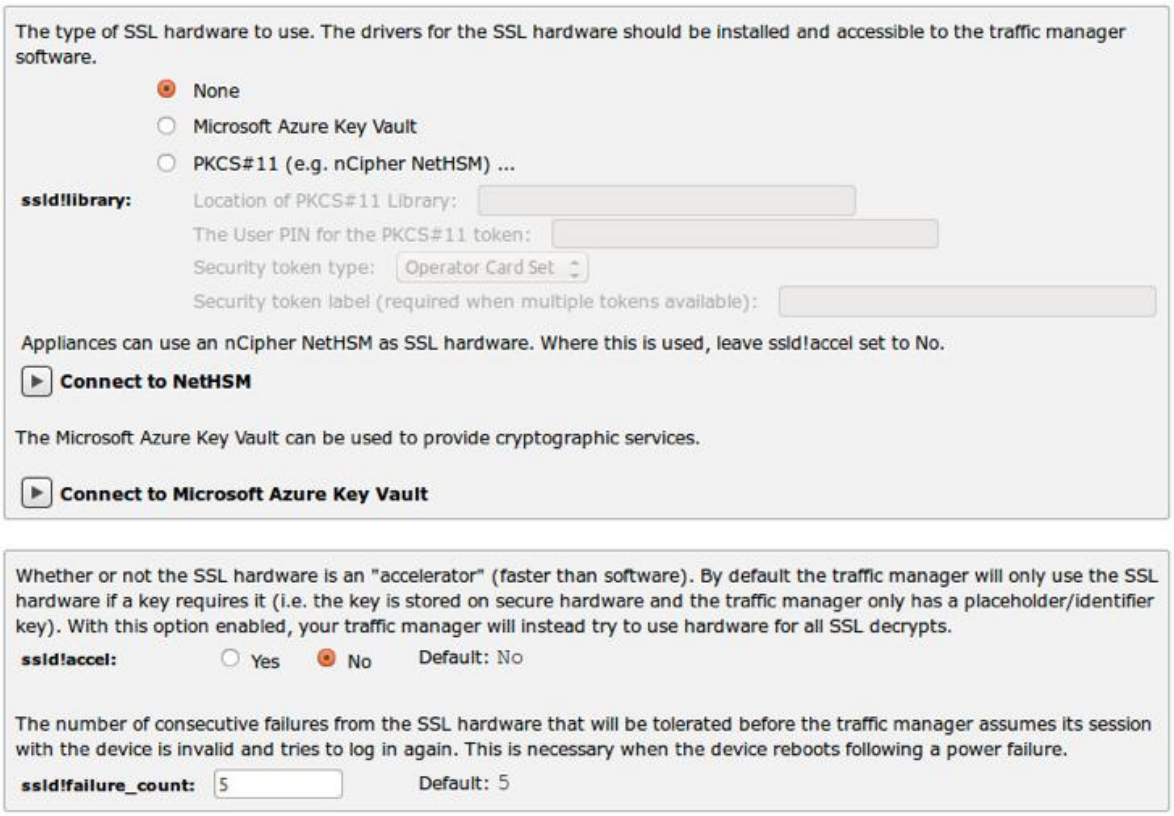

The type of SSL hardware to use. The drivers for the SSL hardware should be installed and accessible to the traffic manager software.
None
Microsoft Azure Key Vault
PKCS#11 (e.g. nCipher NetHSM) ...
ssld!library:
Location of PKCS#11 Library:
The User PIN for the PKCS#11 token:
Security token type:
Operator Card Set
Security token label (required when multiple tokens available):
Appliances can use an nCipher NetHSM as SSL hardware. Where this is used, leave ssld!accel set to No.
Connect to NetHSM
The Microsoft Azure Key Vault can be used to provide cryptographic services.
Connect to Microsoft Azure Key Vault
Whether or not the SSL hardware is an "accelerator" (faster than software). By default the traffic manager will only use the SSL hardware if a key requires it (i.e. the key is stored on secure hardware and the traffic manager only has a placeholder/identifier key). With this option enabled, your traffic manager will instead try to use hardware for all SSL decrypts.
ssld!accel:
Yes
No
Default: No
The number of consecutive failures from the SSL hardware that will be tolerated before the traffic manager assumes its session with the device is invalid and tries to log in again. This is necessary when the device reboots following a power failure.
ssld!failure_count:
5
Default: 5

Fig 44. Settings for using SSL hardware accelerator

The following table describes the settings available:

Setting	Description
ssld!library	Set to "Microsoft Azure Key Vault" to instruct the Traffic Manager to use an Azure Key Vault service. To connect to an Azure Key Vault, click "Connect to Microsoft Azure Key Vault". Set to "PKCS#11" to instruct the Traffic Manager to use generic PKCS compliant hardware such as the Thales e-Security nShield Connect. To connect to a NetHSM device, click "Connect to NetHSM". Additional settings for PKCS#11 are: ▪ **Location**: The system location that the Traffic Manager should search for the PKCS#11 interface library. Leave this field blank for the Traffic Manager to search standard system locations instead. ▪ **User PIN**: The PIN required to create a user session on the SSL hardware. ▪ **Token Type**: The form of security token the Traffic Manager uses to protect the SSL private key: ○ **Operator Card Set**: A physical smart card set with a suitable card reader plugged into the Traffic Manager. The smart card might require an optional user PIN. ○ **Soft Card**: A file on disk that performs the same operation as a physical smart card arrangement. The file requires a user PIN. ○ **Module Protected**: The cryptographic module alone protects the key, with no smart card or user PIN. ▪ **Token Label**: (token types "Operator Card Set" and "Soft Card" only) Use this setting to provide the identifying label or name for the card you are using.

Setting	Description
ssld!accel	Whether to use the SSL device defined on this page for all decryption operations. Set to "Yes" to force the Traffic Manager to use the SSL device for all operations. Set to "No" to mean the Traffic Manager only uses the SSL hardware when the private key is stored securely on it.
ssld!failure_count	The consecutive number of attempts that the Traffic Manager can make before assuming the session is invalid and trying to log in again (due to reboot, power failure, and so on).

Verifying Correct Operation of SSL Devices

The **Diagnose** page displays the results of checks that the Traffic Manager can communicate with the SSL device. Any errors or warnings are indicated.

If you have configured the Traffic Manager to use an SSL device that supports hardware key management, you can then:

- Create new private keys (and corresponding public certificates) on the device using the Traffic Manager. Follow the procedure referred to in "Creating a New Self-Signed SSL Certificate" on page 213 to create a new self-signed Server or Client SSL certificate, making sure to select the "Create private key on hardware" option that is present when a key-management device is configured.
- Manually install private keys and public certificates from the hardware on the Traffic Manager. Follow the instructions for your hardware device to extract an encoded version of the private key and the public certificate. Then install them on the Traffic Manager using the **Import Certificate** tool in the relevant **SSL Server Certificates** and **SSL Client Certificates** catalog.
- Migrate existing private keys from the Traffic Manager onto a newly connected SSL device. To move the key, use the "Migrate certificate to Hardware Security Module" section on the relevant certificate edit page.

  Important: If an attacker has already obtained a copy of your private key, migrating it to an SSL device from the Traffic Manager does not constitute a guarantee that it is now protected. For the highest security, Brocade recommends using keys generated on the SSL device itself.

The Traffic Manager verifies that it can access and use any hardware-managed keys, and indicates an error on the **System > Diagnose** page and the **Event log** if any problems occur.

Note: Although client certificates and corresponding private keys might be marked as *unavailable* on the Diagnose page and in the Event Log, the pool using them cannot be marked as having any problems because the Traffic Manager never knows what client certificate will be asked for until it connects to a back-end SSL node that requires one.

Using the Connect to Microsoft Azure Key Vault Wizard

To connect to an Azure Key Vault, click the "Connect to Microsoft Azure Key Vault" wizard link at **System > Global Settings > SSL Hardware Support**.

However, before you activate the wizard, first obtain the following information:

- **Azure Key Vault URL**: The URL of your designated Azure Key Vault. For example, "https://dev.vault.azure.net".
- **Client ID**: The Identifier (ID) of your user account in the Azure Key Vault.
- **Client Secret**: An alpha-numeric string representing the *password* of your user account.
- **A trusted Certificate Authority**: (Optional) To protect your connections to the Azure Key Vault from "man-in-the-middle" attacks, you can instruct the Traffic Manager to validate the identity of the SSL certificate used by the Azure Key Vault REST API. Obtain the trusted Certificate Authority that signed Microsoft's SSL certificates and store it in the Traffic Manager CA catalog prior to running the wizard.

For help obtaining this information, contact your support provider or see the Microsoft Azure Key Vault documentation at: http://azure.microsoft.com/en-us/documentation/services/key-vault/.

In common with other Traffic Manager wizards, the "Connect to Microsoft Azure Key Vault" wizard runs through a number of steps, each one providing a **Back** and **Next** button to navigate between them:

1. Introduction.

2. Enter the URL of your Azure Key Vault:

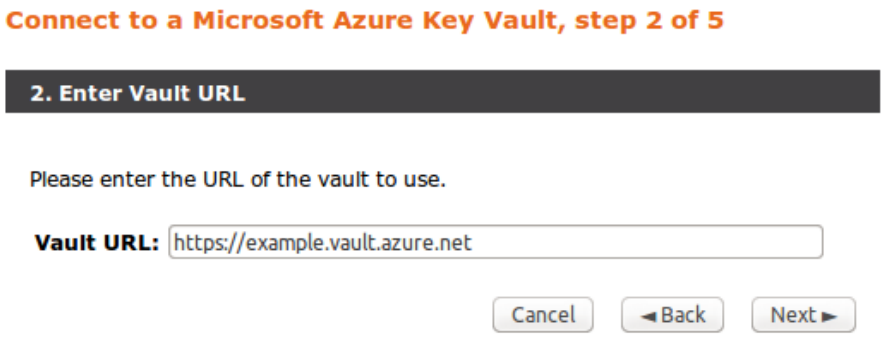


Fig 45. Enter your Azure Key Vault URL

3. Enter your client credentials:

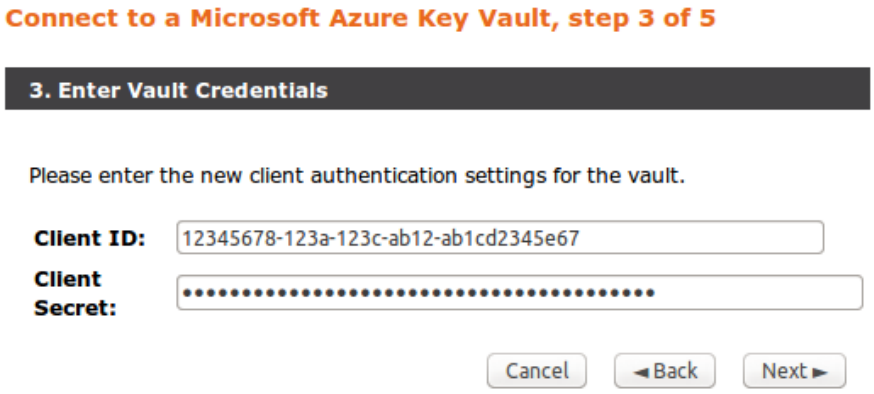


Fig 46. Enter your client credentials

4. Select **Yes** to instruct the Traffic Manager to verify the SSL certificate of the Azure Key Vault REST API, or select **No** to disable verification:

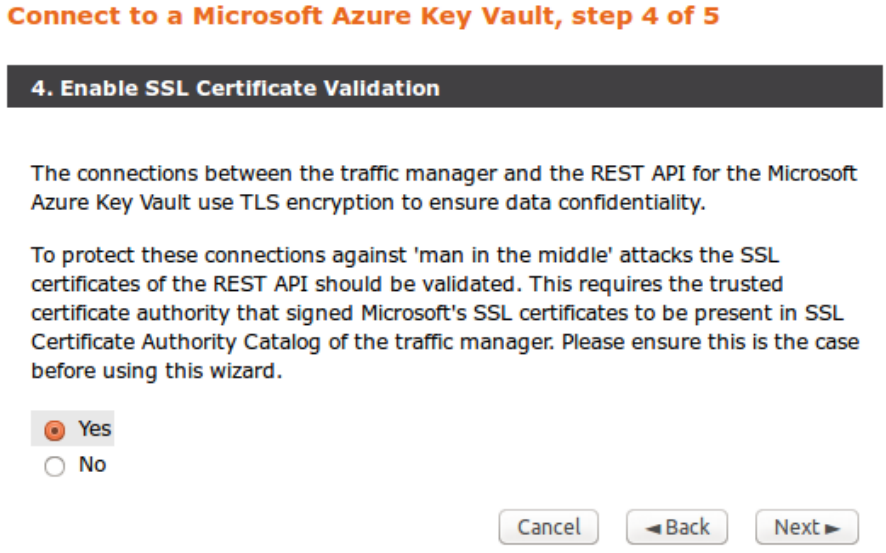

Connect to a Microsoft Azure Key Vault, step 4 of 5
4. Enable SSL Certificate Validation
The connections between the traffic manager and the REST API for the Microsoft Azure Key Vault use TLS encryption to ensure data confidentiality.
To protect these connections against 'man in the middle' attacks the SSL certificates of the REST API should be validated. This requires the trusted certificate authority that signed Microsoft's SSL certificates to be present in SSL Certificate Authority Catalog of the traffic manager. Please ensure this is the case before using this wizard.
Yes
No
Cancel
◄Back
Next►

Fig 47. Verify the Azure Key Vault REST API SSL certificate

5. Click **Finish** to connect to the Azure Key Vault using the settings shown.

If the Traffic Manager is successful, the SSL Hardware Support section updates to indicate the active Azure Key Vault connection.

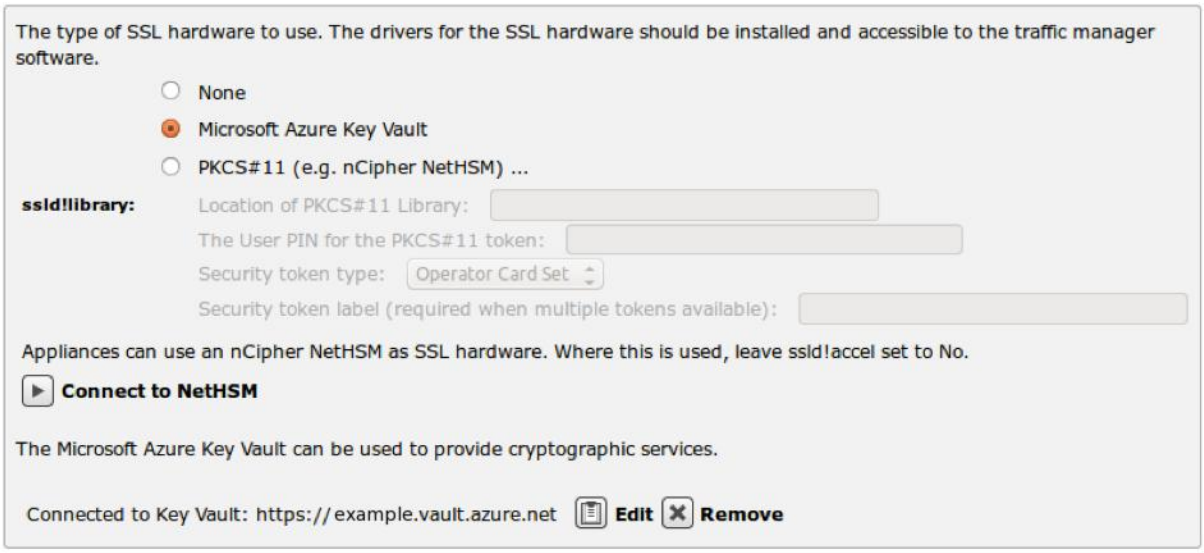

The type of SSL hardware to use. The drivers for the SSL hardware should be installed and accessible to the traffic manager software.
None
Microsoft Azure Key Vault
PKCS#11 (e.g. nCipher NetHSM) ...
ssld!library:
Location of PKCS#11 Library:
The User PIN for the PKCS#11 token:
Security token type:
Operator Card Set
Security token label (required when multiple tokens available):
Appliances can use an nCipher NetHSM as SSL hardware. Where this is used, leave ssld!accel set to No.
Connect to NetHSM
The Microsoft Azure Key Vault can be used to provide cryptographic services.
Connected to Key Vault: https://example.vault.azure.net
Edit
Remove

Fig 48. A connected Azure Key Vault

Click **Edit** to rerun the wizard with alternative settings, or click **Remove** to disconnect the Traffic Manager from the active Azure Key Vault.

Connection failures, warnings, and errors relating to Azure Key Vault are reported on the **Diagnose** page and in the Event Log.

Using the Connect to NetHSM Wizard

Note: This section applies only to Traffic Manager appliance variants.

The Traffic Manager provides a wizard to securely enable communications with a Thales nShield Connect or nCipher netHSM network attached HSM. This wizard simplifies and encapsulates the process into a number of basic steps.

Note: The nShield Connect supersedes the nCipher netHSM device. References to netHSM throughout the wizard should be taken as being synonymous with both. Brocade currently provides support for these devices only. Refer to your support provider for further information.

Certain information is required as pre-requisites to running the wizard. See the nShield Connect documentation for how to obtain these items:

- The IP address of the nShield Connect device with a properly configured Security World.
- The IP address of the Remote File System that the nShield Connect is using.

- An Operator Card Set, with a card inserted into the nShield Connect, and its passphrase.
- One or more nShield Connect-created keys of type "PKCS#11", protected by this Card Set (not module-protected), that you wish to use with the Traffic Manager, including the files produced by the nShield Connect software for use by external applications.

Prior to starting the wizard, you should ensure you have entered the correct PKCS#11 token PIN that corresponds to the smart card used with the nShield Connect. On the **System > Global Settings** page of the Admin UI, locate the *SSL Hardware Support* section. Click the radio button next to the PKCS#11 option in the **ssld!library** config key, and enter the PIN in the box provided. No other settings are normally relevant here, although your specific requirements may vary.

To launch the wizard, click **Connect to NetHSM**.

The wizard runs through a number of steps, each one providing a **Back** and **Next** button to navigate between them.

The four steps are:

1. Introduces the wizard.
2. The license agreement required to use the Thales/nCipher Support Software on the appliance.
3. Enter the IP addresses of your netHSM and its Remote File System.
4. Confirm the identity of the netHSM, by checking the netHSM's reported ESN and HKNETI against its front panel. This ensures that the Traffic Manager is really connected to the intended netHSM, and not to a malicious third party. While visiting the front panel, you should also ensure that the appropriate IP address of the Traffic Manager appliance has been added to the netHSM as an allowed client.

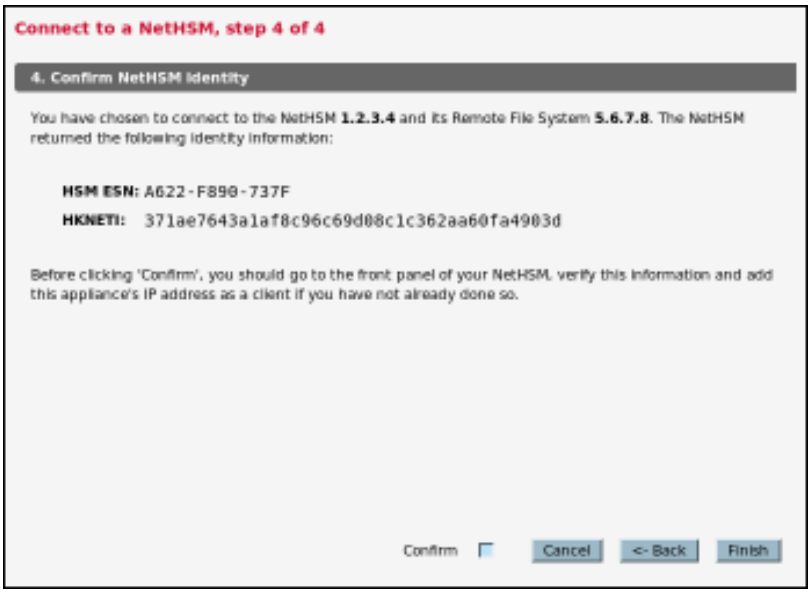

Connect to a NetHSM, step 4 of 4
4. Confirm NetHSM Identity
You have chosen to connect to the NetHSM 1.2.3.4 and its Remote File System 5.6.7.8. The NetHSM returned the following identity information:
HSM ESN: A622-F890-737F
HKNETI: 371ae7643a1af8c96c69d08c1c362aa60fa4903d
Before clicking 'Confirm', you should go to the front panel of your NetHSM, verify this information and add this appliance's IP address as a client if you have not already done so.
Confirm
Cancel
<- Back
Finish

Fig 49. Step 4 of the Connect to NetHSM wizard

5. Select the **Confirm** tick-box and click **Finish** to apply the settings. If all is present and correct, a success page is presented. Alternatively, connection failures or other errors are reported as shown:

Connect to a NetHSM, failed
Wizard failed
The wizard failed to apply the new configuration.
The following problems occurred while finishing the wizard:
PKCS#11 could not connect to hardware
This error was reported by all traffic manager machines.
Cancel
<- Back
Finish

Fig 50. Reporting of connection failures

Typical failures include:

- **PKCS#11 PIN incorrect**: The PKCS#11 User PIN on the **System > Global Settings** page is incorrect.
- **PKCS#11 could not connect to hardware:** The nCipher PKCS#11 library failed to connect to the netHSM/nShield Connect and did not return any additional information to the Traffic Manager. If you see this in the wizard, since the ESN

and HKNETI were just retrieved successfully, the most likely explanation is that the netHSM is not configured to allow this appliance to connect. If you later see this message in your logs, it is more likely that your device cannot be contacted. Restarting/rebooting the Traffic Manager forces a re-connect attempt that can, in some circumstances, resolve this issue.

- **Failed to synchronize nCipher Remote File System:** The nCipher tools could not retrieve a copy of the Remote File System data from the IP address entered. This can happen if the Remote File System could not be contacted, if it is not associated with the same nShield Connection as the appliance, or if it has been configured not to allow the appliance to connect (or to require an nToken, which a Traffic Manager appliance cannot use).

Upon successful completion of the wizard, the "Connect to NetHSM" link of the **Global Settings** page changes to indicate that the connection has been made:

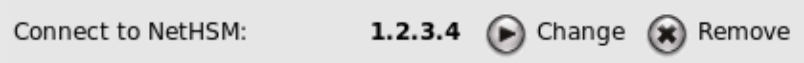


Fig 51. A successful connection to a netHSM device

Identifying Keys and Certificates Stored on a Secure Device

After you have successfully connected to secure hardware or an Azure Key Vault, you can use the key and certificate files from the device as you would any other key pair (including editing the certificate). Provided the device is operating normally, the Traffic Manager delegates all cryptographic operations that require the private key to the device. In the Admin UI, the Traffic Manager identifies keys as being stored on secure hardware, or in the Azure Key Vault, as shown:

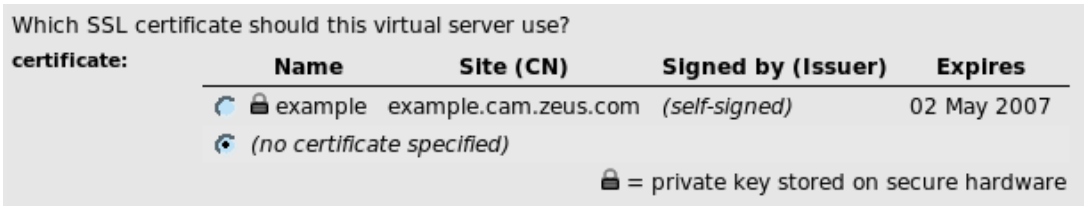


Fig 52. The 'padlock' icon identifies keys stored on secure hardware

CHAPTER 14 Health Monitoring

This chapter describes the heath monitoring capabilities found within the Traffic Manager, which can be used to monitor server nodes for correct operation and raise alerts and route round failed nodes when an error is detected.

Which Nodes Are Monitored?

The Traffic Manager uses two methods to monitor the correct operation of nodes: passive monitoring (checking the status of a node when it is used) and health monitoring (additional tests that are run against a node on a periodic basis).

Passive monitoring tests are only performed against nodes that are actively in use.

Health monitors are executed against all of the nodes in a pool. They are only run against pools that are in use:

- Pools that are configured as the default pool for a virtual server
- Pools that are explicitly referenced by name in a TrafficScript or RuleBuilder rule, either by the `pool.select()` or `pool.use()` functions
- Pools that are configured as the failpool for a pool that is in use

If a pool is not referenced by the Traffic Manager configuration in this way, it is considered to be 'not in use' and is not monitored using health monitors. Any failures of nodes will not be detected.

Note: Pools must be explicitly referenced by name when they are used in a TrafficScript rule. By default, referencing by a variable (`pool.use( $name );`) or any other means is not permitted by the Traffic Manager.

If you enable the setting **trafficscript!variable_pool_use** (in the **Global Settings** page), you may use variables for pool names. If this setting is enabled, the Traffic Manager will execute health monitors against all of the pools you have configured, not just the ones that are clearly in use.

Using Nodes in Multiple Pools

The same node (IP address and port) may be referenced in several different pools. If a node has failed in one pool, it is not used by the load balancing or session persistence decisions. However, that node may be used in other pools until the passive monitors and health monitors assigned to those pools report a failure.

Example

Your Web application has 4 back-end servers (nodes). Each node hosts the same dynamic content and static content. If the dynamic content on a node fails (for example, the java servlet crashes), you might still want to use that node for other static content:

- Create two pools named "Dynamic" and "Static", each containing all 4 nodes.
- Create a rule that uses pool "Dynamic" for dynamic content (Java Servlets, PHP and ASP files etc) and uses pool "Static" for all other content.
- If a node fails in the "Dynamic" pool and fails to send a valid HTTP response, the passive monitoring used by that pool determines that node has failed. No more traffic from the "Dynamic" pool is sent to that node. However, the "Static" pool continues to send traffic to that node, as long as it returns valid HTTP responses for requests from the "Static" pool.
- For more fine-grained detection of errors, you can assign different health monitors to each pool. For example, the health monitors for the "Dynamic" pool can send synthetic PHP or ASP requests. If these monitors fail, the node is considered to have failed in the "Dynamic" pool, but not in other pools.

Passive Health Monitoring

A pool performs a set of checks every time it attempts to send a request to a node; this process is referred to as 'Passive Monitoring':

- The Traffic Manager attempts to connect to a node; if the connection is refused, or is not established within the **max_connect_time** setting (default 4 seconds), the request is considered to have failed.
- The Traffic Manager writes the request data down the connection; if the connection is closed prematurely, or if the beginning of a response is not received within **max_reply_time** seconds (default 30 seconds), the request is considered to have failed.
- **SSL only**: if the SSL handshake to the node fails, the request is considered to have failed.

Note: The **max_connect_time** and **max_reply_time** settings are properties of the **Connection Management** settings (see the *Connection Management* section of CHAPTER 5) in a pool.

Retrying Failed Requests

If these checks fail, the pool *may* try the request against a different working node, and *may* try every node in the pool before abandoning the request. The behavior is determined by the *Idempotent* status of the request.

By default requests are assumed to be idempotent, i.e., they can be safely retried multiple times without undesired side effects. An exception to this is any request received through a virtual server using one of the *generic*-type protocols ("Generic Client First", "Generic Server First", "Generic Streaming"). In order to be idempotent by default, an end-point to the request must first be defined[8] in order for failure to be measured and retries to be triggered.

RFC 2616 defines some HTTP requests as non-idempotent[9] (e.g., they may cause a transaction to take place or change state on the server). The Traffic Manager follows these recommendations and will treat HTTP GET, HEAD, PUT, DELETE, OPTIONS and TRACE methods as idempotent; all other requests are considered non-idempotent.

- **Idempotent (no side effects)** - The Traffic Manager will retry these requests against other believed-to-be working nodes, and may try every node in the pool before abandoning the request.
- **Non-idempotent (side effects)** - The Traffic Manager will only retry a non-idempotent request if it failed to open a TCP connection to the failed node.

Note: When the Traffic Manager establishes a TCP connection, it immediately writes the request data down that connection. The Traffic Manager is not able to determine whether or not the node has received the request data and begun processing it. Therefore, non-idempotent requests are only retried if the connection could not be established in the first place.

You can override the idempotent/non-idempotent decision made by the Traffic Manager by using the `request.setIdempotent()` and `http.setIdempotent()` TrafficScript functions to indicate to the Traffic Manager that a particular request should be considered safe to retry.

503 Server Errors

503 Server Error responses are treated differently, because you typically would not wish to try a request that generated a server error against every node in your cluster.

A request that generates a 503 Server Error is only retried if:

---

[8] Achieved via TrafficScript, using functions such as `request.endsWith()` or `request.endsAt()`

[9] http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

- Passive monitoring is enabled.
- No session persistence was in place.
- The request is idempotent (no side effects).

A request is tried a maximum of three times before the 503 error is sent back to the client.

Node Failures

The Traffic Manager will infer that a node has failed if connections to that node fail consistently, with **node_connection_attempts** (default 3) failures in a row with no intermediate successful transactions.

Once a node has been deemed to have failed, it is not used for at least **node_fail_time** seconds (default 60 seconds), after which it is tentatively used to determine if it has recovered. However, where *all nodes* in the pool have failed, the Traffic Manager will immediately send traffic to a recovered node regardless of the **node_fail_time** setting.

Note: The **node_connection_attempts** and **node_fail_time** settings are found on the **Pool > Connection Management** page.

You may wish to try enabling the **log!server_connection_failures** setting in the **Connection Error Settings** section on the **Virtual Server > Connection Management** page. This could help provide useful log information regarding what the actual node failures were. See the *Handling Errors* section of CHAPTER 4 for more details.

Enabling and Disabling Passive Monitoring

Passive Monitoring is used by default, but can be disabled on a per-pool basis using the **passive_monitoring** setting in the **Monitors** section of a pool configuration. If this setting is disabled, you should ensure there are suitable **Health Monitors** configured otherwise failed requests will not be detected and subsequently retried.

Overview of Health Monitors

In addition to the inferences from passive monitoring, a pool may be configured with explicit **Health Monitors** that perform periodic tests and verify the correct operation of each node. By using health monitors, the Traffic Manager can detect failures even when no traffic is being handled.

The tests are performed at configured intervals against each node in a pool; if they fail sufficiently often, that node is marked as unavailable, and the Traffic Manager

will stop sending traffic to that node. Pool-wide monitors just test a single machine and can be used to indicate that an entire pool is down.

The monitors are held in the Monitors Catalog and can be applied to any pool. Each performs a specific test. These range from simple tests, such as pinging each node, to more sophisticated tests which check that the appropriate port is open and the node is able to serve specified data, such as your home page.

The Monitors Catalog

Setting Up a Monitor

To access the Monitors Catalog, click the **Catalogs** button on any page of the Admin Server interface. You can unfold the **Monitors Catalog** section to see the monitors you have already set up. Clicking on **Monitors Catalog** takes you into the catalog.

The Traffic Manager contains a number of preset monitors. Some of these deal with generic TCP client-first or server-first protocols, and others are protocol-specific. For instance, the built-in **Full HTTP** monitor requests a specified page from a Web server and looks for a suitable status code in the response.

All monitors have the following configurable settings:

- **delay** - Monitors are executed repeatedly against each node. Once a monitor has completed or times out, the Traffic Manager pauses for this configured period of time before trying the monitor again.
- **timeout** - If a monitor fails to complete within this timeout period, it is aborted and judged to have failed.
- **failures** - A monitor may occasionally fail for unpredictable and unrepeatable reasons, such as a brief network glitch. This setting indicates how many times a monitor must fail consistently before the monitored node is judged to have failed.
- **verbose** - This setting enables verbose monitor logging in the error log. It is useful for testing a monitor, but should not be used in a product system or for a long period of time because it can fill up the error log file.

You can edit the built-in monitors: click the name of one, and modify the timings, failure conditions, and logging, along with any other monitor-specific settings (such as the Web page to request). Click **Update** to commit your changes.

You can also click **Copy Monitor** to copy the built-in monitor under a new name, and make your changes to the copy.

Built-in Health Monitors

The Traffic Manager contains a number of predefined monitors in the **Monitors Catalog**. These built-in monitors perform standard tests for various protocols.

Basic Monitors

Basic Monitors perform simple tests against a node. For example:

- **Ping**: The ping monitor performs ping requests against each node. It will fail if no response is received.
- **Connect**: This performs TCP connects to the node. Its purpose is to ensure that a server is listening on the port, and will accept traffic. This monitor can cause some programs (such as exim) to output errors; in this case a more protocol specific monitor should be used.

Advanced Monitors

Advanced monitors perform more complex request and response tests against a node. These monitors are used as building blocks for other monitors, and can use SSL to connect and transfer the request and response. For example:

- **Client First**: This is a monitor based on the TCP transaction monitor. It will send some data to the node, and match the response against the supplied regular expression. In order to use it, the data to send (the **write_string**) must be configured.
- **Server First**: This monitor can be used to check services where the server writes data first. It will connect to the node, and check that the server returns some data. This monitor can easily be altered to check for specific data being returned by the server.
- **Full HTTP / HTTPS**: This monitor can be used to perform HTTP or HTTPS requests. The host header, URL and authorization parameters can all be configured.

The response return code is matched against the value in **status_regex**. For example, if you are expecting a 200 return code, set **status_regex** to "^200$". Alternatively, use "^[234][0-9][0-9]$" to accept errors from 200 to 499.

The response body is matched against the optional setting **body_regex**.

If the pool that an SSL monitor is assigned to has **ssl_server_name** enabled, the monitor sends a Server Name Indication (SNI) TLS extension with the name of the node to the back end.

Protocol-Specific Monitors

Protocol-specific monitors perform specific tests against a node. These include:

- **DNS**: This is an example of a program monitor. It will run a program (`ZEUSHOME/zxtm/conf/scripts/dns.pl`) that makes a DNS request to the node, and validate the address returned.
- **FTP**: This monitor is based on the TCP transaction monitor. It will make a simple FTP connection, and check that the server is responding correctly. It only checks that the banner is returned by the server, and does not authenticate.
- **POP**: This monitor is based on the TCP transaction monitor. It will make an initial connection to the node, and check that the correct POP banner is returned.
- **SMTP**: This is identical to the POP monitor, except that it checks for the SMTP server banner being returned.
- **Simple HTTP and HTTPS**: These monitors request the home page from each node. They check that the response code is a 2xx, 3xx, or 4xx response code, and indicate a failure of not.
- **RTSP**: This monitor will send an RTSP DESCRIBE request to the server. It will check for a 2xx, 3xx or 4xx response or report a failure if one is not received. Note that some RTSP servers require the presence of a path in such a DESCRIBE request, in which case you should enter one in the rtsp_path setting. The protocol specific additional settings for RTSP are:
  - **rtsp_path**: the path to be used in the DESCRIBE request. This is the path to the multimedia file being streamed.
  - **rtsp_status_regex**: The status code regular expression that the RTSP response must match.
  - **rtsp_body_regex**: The regular expression that the RTSP response body must match.
- **SIP (UDP)**: This monitor will send a SIP OPTIONS request to the server with Max-Forwards set to 0. It will check for a 2xx, 3xx or 4xx response and report a failure if one is not received.
  - **sip_status_regex**: The status code regular expression that the response must match.
  - **sip_body_regex**: The regular expression that the SIP response body must match.
  - **udp_accept_all**: Should this monitor accept responses from any IP and port?
- **SIP (TCP)**: This monitor uses the same request as the SIP-UDP monitor but sends it over a TCP connection.

Custom Health Monitors

The main **Monitors Catalog** page has a **Create New Monitor** section, where you can create an entirely new monitor. There are several options for the underlying type:

Monitor Type	Description
TCP Transaction	This performs a TCP transaction with the target machine, with an optional string of data to write to the connection It can look for a specified regex in the response.
Ping	This pings the target machine at specified intervals.
External Program	This runs an external program, whose file path on the Traffic Manager machine you specify. It sends two built-in arguments, `port` and `ipaddr`, along with any others specified. An exit code of 0 is classed as a success; a code other than 0, or a timeout, is a failure. *ZEUSHOME*`/zxtm/conf/scripts` contains your custom monitor executables; you can see an example there.
HTTP	This sends an HTTP request to the target server, optionally using SSL, with specified parameters such as host header and the URL path to use. It searches for a status code regex in the response code, and a regex against the response body data.
TCP Connect	This makes a TCP connection with the target machine, to check that the appropriate port is open.
RTSP	This issues an RTSP DESCRIBE request to the target server, with the specified rtsp_path parameter. It looks for a status regex in the response, and a regex against the response body data.
SIP	This sends a SIP OPTIONS request to the target server using the specified transport protocol. It searches for the status regex in the response code and the body regex in any body of data that was received. A failure is reported if either of these do not match.

When you have chosen the basic settings for your monitor, click **Add Monitor**. You can click the name of the monitor to edit it.

It is possible to build sophisticated custom monitors, using either the **TCP Transaction** or **HTTP** monitor templates. For example, a pair of HTTP-based monitors could test the secure part of a Web site to verify that:

a) A request with a valid username and password receives a valid 200 response.

b) A request with an incorrect username and password receives a 401 or 403 error response.

TCP Transaction monitors are useful building blocks for nodes that use other protocols.

For more sophisticated tests, a monitor can run an external program or script. **External Program monitors** are described in the *External Program Monitors* section of CHAPTER 14.

Per-Node and Pool-Wide Monitors

Monitors fall into two categories: ***per-node*** and ***pool-wide***. A per-node monitor tests the health of each node in the pool. A pool-wide monitor performs tests on one machine, which influences the health of the entire pool. For example, a mail server pool might keep its data on an NFS server, which each of your back-end servers accesses. A pool-wide monitor could test this server. If it fails, none of the back ends can retrieve the data so the whole pool is deemed to have failed.

You can choose whether your monitor should be per-node or pool-wide. If it is pool-wide, you must specify a machine (and possibly port) for the monitor to test, such as an NFS server used by the pool.

Using Health Monitors

Applying a Monitor to a Pool

To apply a monitor to a pool, click the **Services** button and then the **Pools** tab. Click the name of a pool to edit it.

Click Health Monitoring to choose the monitors used by the pool. You can add a new monitor from the drop-down list, clicking **Add Monitor** when you have finished. To change settings for a monitor attached to your pool, click the **Edit In Catalog** link against it.

Note: A monitor sending test requests will increase the load on a pool, albeit by a small amount. You should ensure that your back-end servers can handle the traffic produced by the monitors, to avoid a DoS failure.

External Program Monitors

An external program monitor can be written in any language. The Traffic Manager passes command-line arguments to the executable:

```
domonitor --ipaddr=<machine_to_monitor> \
          --node=<nodename> \
          --port=<port_to_monitor>
```

Note: The `ipaddr`, `node` and `port` arguments are always passed to an external program monitor. The `node` argument is the hostname part of the node, as configured in the pool, and the `ipaddr` is the corresponding IP address.

You can also specify additional arguments for the monitor, which may be useful if you are using the same executable for several different monitoring tasks. Use the **Program Arguments** section of the **Edit** page for the monitor in question.

For example, if you configured two additional arguments (named 'regex' and 'interface') and gave then appropriate values on the Edit page, the monitor executable would be invoked as follows:

```
domonitor --ipaddr=<machine_to_monitor> \
          --node=<nodename> \
          --port=<port_to_monitor> \
          --interface=<interface_value> \
          --regex=<regex_value>
```

A successful monitor should exit with an exit code of 0; a timeout or non-zero exit code is interpreted as failure, and in this case the contents of STDERR are written to the Event Log.

When you develop and test a monitor executable or script, you can easily run it from the command line, providing arguments in the format described above. Your monitor can emit debugging information to STDOUT.

When the monitor is used by the Traffic Manager, you can enable verbose mode. If the monitor is called in verbose mode, anything printed to STDOUT by the

executable is written to the Event Log; verbose mode can be toggled on the monitor's Edit page.

Uploading Monitors to the Traffic Manager

You should use the **Catalog > Extra Files > Monitor Programs** page in the Traffic Manager Administration Interface to upload, manage and delete monitors on the Traffic Manager:

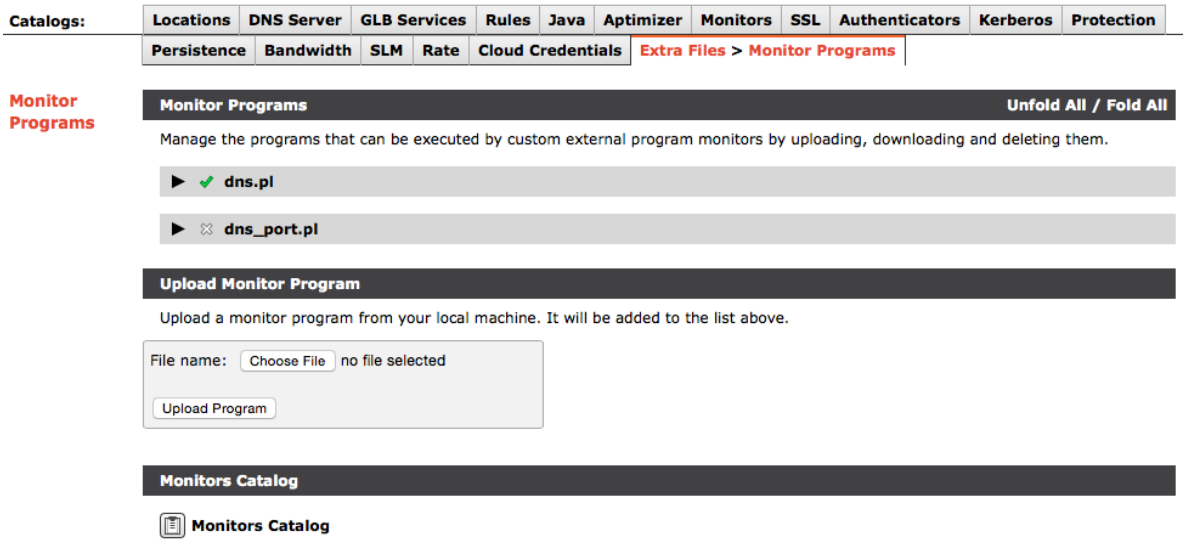


Fig 53. Managing External Monitor programs

Writing Monitors in Perl

If you choose to write your monitor in Perl, the Traffic Manager provides a helper module, `Monitor.pm`, that is included with your Traffic Manager. This includes a `ParseArguments()` function, as well as functions to exit with success or failure and to write out logs.

```
use Zeus::ZXTM::Monitor qw( ParseArguments
                            MonitorWorked
                            MonitorFailed
                            Log );

%ARGS = ParseArguments();

Log( "Running test" );
if( do_test( $ARGS{ipaddr}, $ARGS{port} ) ) {
    MonitorWorked();
} else {
    MonitorFailed( $error_message );
}
```

For more information on `Monitor.pm`, use this command:

```
$ perldoc ZEUSHOME/zxtm/lib/perl/Zeus/ZXTM/Monitor.pm
```

CHAPTER 15 Service Protection

This chapter explains some of the risks associated with Internet hosting, and how to use the Traffic Manager to mitigate those risks to your services.

Classes of Risk

Denial of Service (DoS)

A ***Denial of Service*** or ***DoS*** attack is characterized by a malicious attempt to prevent legitimate use of a service. This could take the form of attempting to flood a network, or disrupt connectivity between machines. It may be designed to consume the resources normally available for specific users or applications, for example exhausting the available CPU, bandwidth or storage of a server or cluster of servers.

DoS attacks are based around exploiting design weaknesses, flaws in the operating system, or services with unbounded access to a resource such as CPU time, disk or memory. For example, a malicious attacker may find a way to craft a request to a database-driven Web site that requires extremely intensive SQL operations to complete. This could effectively deny access to the database for the duration of the query.

DoS attacks that are delivered through exposed services such as websites are the most common class of risk. The Traffic Manager is designed to assist in mitigating the effects of such attacks, or preventing them altogether.

Web Worms and Viruses

Vulnerabilities in operating systems have led to many Internet ***worms*** which propagate by installing themselves, using the weakness as a point of entry. For example, a worm can install a payload sufficient to propagate itself further. Worms also affect desktop PCs. The frequency of this class of attack is increasing, and is often part of a planned DDoS attack in which the compromised machine will play a part.

In addition to worms, users of desktop PCs are exposed to risks from viruses, often received by email or from unscrupulous websites. These viruses may install services on the PCs which can be controlled remotely.

Distributed Denial of Service Attacks (DDoS)

Distributed Denial of Service attacks use large numbers of client machines, often "recruited" after being compromised by worms or viruses. A malicious attacker can

control these clients remotely, making the combined impact of a focused attack on an individual service provider or business far greater than a conventional DoS attack. Mitigating distributed attacks is far more challenging than handling a DoS attack, as it requires the cooperation (e.g. outbound packet filtering) of other ISPs to fully address the issue.

The Traffic Manager can provide protection against basic DDoS attacks as well as DoS attacks, provided network bandwidth is not flooded and provided there is sufficient capacity to manage the incoming connections.

Malformed HTTP Attacks

Even correctly firewall-protected Web servers must still present their public services on port 80 to the Internet. HTTP's client-server architecture means that clients can send data to these Web servers freely through port 80. This information can be crafted to maliciously overload or subvert a server. Attacks exist where the request is malformed in such a way as to exploit bugs and compromise the Web server to give control to a remote attacker.

Firewalls and Other Security Measures

Note that the Traffic Manager is not a firewall; it is intended to be used in conjunction with a dedicated firewall. See CHAPTER 23 for a full discussion of secure operation.

Protection Features

The Traffic Manager is a robust solution that is not affected by known classes of exploits. For example, they are believed to be invulnerable to all known worms and malformed HTTP attacks. In addition, since the Traffic Manager is placed at the point of ingress for traffic to your clusters - and is able to make intelligent routing decisions - it can help defend your platform from obvious network attacks and filter malicious content transparently. This is particularly effective if your back-end servers use a private network rather than having Internet IP addresses.

The Traffic Manager is also equipped with ***Service Protection*** features, which can be configured for each virtual server independently. These features allow requests to be managed and filtered dynamically in a number of ways.

Network Access Restrictions

The source IP address of a client can be used to decide whether or not to accept requests. If a particular IP address or network block is generating malicious requests,

the Traffic Manager can be configured to drop all connections efficiently from these addresses, thus protecting the back-end pools from attack.

Connection Limiting

Often, clients abusing a system generate abnormally large numbers of connections. The Traffic Manager is able to detect and filter unusually heavy activity, based on the traffic from the top 10 busiest IP addresses or on the number of connections per minute being made from each IP address. This allows the Traffic Manager to deny access to clients making overly intensive requests to your systems.

Malformed HTTP Filtering

The Traffic Manager is able to detect and reject certain classes of malformed HTTP requests, and enforce standards-compliant requests from clients. For instance, the Traffic Manager can filter binary data from requests and prevent very large headers being used as a vector for a DoS attack.

Rule-Based Protection

The Traffic Manager service protection system can be configured to screen all incoming requests so that any matching the specified criteria are dropped. This functionality can be used to protect your system against known vulnerabilities in third-party applications running on your websites. For example, some business-critical applications such as shopping carts are sometimes found to have insecurities that are triggered by requesting a URL containing badly formed parameters.

The forms of these requests are often made public on security mailing lists. The Traffic Manager service protection system enables you to use this information to filter out any requests containing these badly formed parameters, so that you can resolve these vulnerabilities quickly and easily across all your websites.

You can configure the Traffic Manager to block requests by building up a list of rules that are used to filter all incoming requests. In addition you can limit the sizes of requests and the format of the information they can contain.

Enabling Service Protection

Service Protection is configured in two stages. Firstly, ***service protection classes*** are created in the **Catalog**. You can configure as many service protection classes as you require, each with its own settings and rules.

Secondly, the service protection class is assigned to one or more virtual servers, thus associating your protection settings with requests for that virtual server. You can assign the same service protection class to many different virtual servers.

Adding a Service Protection Class

In order to add a service protection class, first click the **Catalogs** button. Select **Service Protection catalog**.

The Traffic Manager now shows you a list of existing service protection classes, and allows you to add a new service protection class by entering a name and clicking **Create Class**.

When you add a new service protection class, there are five main configuration groups.

Basic Settings

These settings enable and disable service protection and dictate in which mode the service protection class operates.

If you wish to bypass this service protection class temporarily, you can enable or disable service protection for the class. This is particularly useful if many virtual servers are configured to use the service protection class.

You can configure the log buffer time, which is the amount of time the Traffic Manager buffers logging in order to reduce disk writes (and reduce any knock-on problems if heavy logging results during a DoS attack). The Traffic Manager also includes a debug option, which causes verbose output to be logged.

Important: The debug option can create heavy loads on your Traffic Managers, and should not be used in production systems.

The Traffic Manager also offers a test mode, whereby the decisions made by the Traffic Manager with this service protection class are logged, but not acted upon. This option is designed to help tune the parameters of the service protection class to suit your services, and should be used for a period of testing before switching on protection.

Simultaneous Connections

These settings specify limits on the number of connections that the Traffic Manager allows from individual IP addresses:

- **max_1_connections**: The maximum number of simultaneous connections the Traffic Manager allows from each connecting IP address. Set to 0 to disable this limit.

- **per_process_connection_count**: Determine whether the Traffic Manager uses a per-process simultaneous connection count model (each Traffic Manager typically has several processes, one process per available CPU core).

  Click "Yes" to allow the connecting IP address to make up to **max_1_connections** connections to each process within the Traffic Manager.

  Click "No" to allow the connection IP address to make up to **max_1_connections** connections to the Traffic Manager as a whole.

  If **per_process_connection_count** is set to "Yes", the following additional limits apply:

  - **max_10_connections**: The maximum number of simultaneous connections from the top 10 busiest connecting IP addresses combined. The top 10 clients typically represent the majority of the load on a particular virtual server if any form of DoS attack is in progress or other abusive connections are being made to the Traffic Manager. It is therefore desirable to limit this particular group of users while leaving other requests untouched. For normal use, Brocade suggests a value between 1 and 10 times the **max_1_connections** limit. To remove the effect of this limit, set it to at least 10 times the **max_1_connections** limit. The Traffic Manager ignores this setting if **per_process_connection_count** is set to "No", or **max_1_connections** is 0, or **min_connections** is 0.
  - **min_connections**: The entry threshold for the **max_10_connections** limit. The **max_10_connections** limit is not applied to connecting IP addresses with this many or fewer simultaneous connections. Set to 0 to disable both the **max_1_connections** and **max_10_connections** limits. The Traffic Manager ignores this setting if **per_process_connection_count** is set to "No".

Connection Rate

Important: Connection rate limiting is per process, and each process independently imposes the rate on the connections it is handling.

These settings allow you to limit the rate at which each connecting IP address might make new connections or requests:

- **max_connection_rate**: The maximum number of connections that any individual IP address can make to the Traffic Manager during the interval specified in **rate_timer**. Set to 0 to disable the limit.

  Note: In the case of HTTP, each individual request is counted as a connection, even if they are tunneled within a single Keepalive connection.

- **rate_timer**: The interval (in seconds) the Traffic Manager uses to determine how frequently **max_connection_rate** is assessed. For example, a value of 60 imposes a limit of **max_connection_rate** connections per minute. All connection attempts above this limit are rejected for the remainder of the **rate_timer** interval.

For more fine grained connection rate control, use the Request Rate Shaping feature described in CHAPTER 17, "Request Rate Shaping".

Access Restrictions

The **Allowed** list allows you to specify allowed IP addresses or networks that are exempt from the normal service protection checks in this service protection class. In other words, IP addresses in this list are not subject to any of the limits set in the Simultaneous Connections and Connection Rate sections.

Use the **Banned** list to declare IP addresses or networks from which the Traffic Manager must always drop connections.

When checking a source IP, the subnets in both lists are evaluated using a longest-match first algorithm. In other words, the more specific subnet takes precedence. It is therefore possible to ban a more specific subnet contained within a less specific allowed one and vice versa. For example, allowing 192.168.0.0/16 and banning 192.168.128.0/24 allows requests from IP 192.168.0.137, but not from 192.168.128.137. If the same subnet is present in both the **Allowed** and the **Banned** list, requests from that subnet are allowed.

To add to either list, input the IP or network address in the appropriate box. Use one of the following formats:

- `192.0.2.0/255.255.255.0`
- `192.0.2.0/24`
- `192.0.2.1`
- `192.0.2.`

HTTP-Specific Settings

These settings provide extra checks which can be performed on HTTP traffic.

The Traffic Manager allows you to enforce compliance with RFC2396; this checks that the URL is properly formed according to accepted guidelines. Since standard Web browsers comply with RFC2396, this test will reject requests that include unusual or malicious data or formatting, while allowing legitimate requests to pass through.

The length of the HTTP body and any HTTP header can be restricted. This prevents a malicious attacker sending a very large or endless body or header. Web servers normally buffer incoming requests in progress in memory, and can therefore be vulnerable to a memory depletion attack. You can guard against this by preventing clients from sending large amounts of request data.

You can also specify a limit to the total size of all the HTTP headers. This prevents an attacker from constructing a very large request using a large number of HTTP headers.

You can limit the size of the URL requested, to prevent buffer overflow attacks and other attempts to use the URL as a vector for large amounts of data or worms.

After decoding, the Traffic Manager can reject HTTP headers that contain binary data. This is often an indicator of an attempted buffer-overrun attack, although some poorly written applications use binary data. If you suspect that your applications could be affected, you should first prove this feature using the test mode.

Lastly, the Traffic Manager allows you to specify whether to reject requests silently if they fail one of the above tests, or to send HTTP error messages to the client that makes the request. Sending error messages incurs slightly higher overheads, which can be significant if a DoS attack is in progress.

Note: If you associate this service protection class with a virtual server that does not handle HTTP traffic, then these settings will be ignored.

Service Protection Rule

This section displays the current rule, if any, which the service protection class is using. If you wish to change the rule, select the rule you wish to use from the drop-down menu. For more information on TrafficScript, please see CHAPTER 8.

To commit your settings and create the service protection class, click **Update**.

Applying a Service Protection Class to a Virtual Server

In order for a service protection class to take effect, it must first be assigned to a virtual server. To do this, click the **Services** button. Next, click the **Virtual Servers** tab and choose an existing virtual server.

Select **Service Protection**. The user interface offers a drop-down box of the service protection classes that are available in the catalog.

Once you have selected a service protection class, click **Update.**

The Traffic Manager applies the new service protection class immediately across all Traffic Managers. A summary table indicating whether the service protection class is in test or debug mode is shown. If the service protection class has not been enabled (see the *Basic Settings* section of CHAPTER 15) then this is also indicated.

Service Protection Performance

The Traffic Manager's service protection system requires additional memory and CPU time for each request. If you are managing a large number of requests and use sophisticated rules in your service protection class, every request for a virtual server using this service protection class will take slightly longer. For large systems it may be necessary to add further Traffic Managers to your front-end network in order to provide additional capacity.

CHAPTER 16 Bandwidth Management

This chapter explains what Bandwidth Management is, and how to configure your Traffic Manager to control how bandwidth is used on your network.

Note: Bandwidth Management is not available on all Traffic Manager variants. If required, it can be obtained via a software or license key upgrade.

What Is Bandwidth Management?

Bandwidth management allows the Traffic Manager to limit the number of bytes per second used by inbound or outbound traffic, for a virtual server or by the type of request.

Normally, network bandwidth is provided at the highest rate possible for all connections. This may result in uneven use of your network, possibly with too much bandwidth being used by secondary services at the expense of your most critical services. Bandwidth management with the Traffic Manager allows you to control this imbalance explicitly.

For example, you may have a 20Mbits/s network connection which is being over-utilized for FTP downloads, which is affecting the responsiveness of the main HTTP service. You may therefore wish to limit the bandwidth to the FTP virtual server to 2Mbits/s.

Bandwidth limits are automatically shared and enforced across all the Traffic Manager machines in a cluster. Individual Traffic Manager machines take different proportions of the total limit, depending on the load on each, and unused bandwidth is equitably allocated across the cluster depending on the need of each machine.

Note: Bandwidth management is only applicable to traffic sent by the Traffic Manager. In other words, the Traffic Manager can control the bandwidth used when writing requests to server nodes, and when writing responses back to clients. The Traffic Manager CANNOT control how quickly clients write the requests to the Traffic Manager system, or how quickly servers attempt to write responses to the Traffic Manager.

Bandwidth management only works on data sent to external addresses. It does not apply to data transferred internally within the Traffic Manager, such as with configurations where the data is passed from one pool to another internal virtual server. In these cases, the host system will use the system's loopback network interface which is not subject to bandwidth restrictions.

Configuring Bandwidth Management

The Traffic Manager's bandwidth management features are provided via the Catalog. This means that you may add more than one Bandwidth Management Class, each with its own limits. It also means that the choice of Bandwidth Management Class can be made through a TrafficScript rule, providing greater flexibility for situations where it is helpful to distinguish between requests to the same virtual server. For example, the Traffic Manager's Bandwidth Management system can be used to apply different limits for CGI requests than for image requests (see examples, below).

If you have a cluster of more than one Traffic Manager, the Bandwidth Management system takes this into account. You do not need to make adjustments when new servers are being added, as this is done automatically.

Adding a Bandwidth Class to the Catalog

To add a Bandwidth Management Class to the Catalog, click **Catalogs** and then click **Bandwidth** in the menu bar.

The Traffic Manager now shows a list of the existing Bandwidth Management Classes, if any, and provides a box where you can create a new Bandwidth Management Class.

Type a name into the box and click **Create Class**. The Traffic Manager now displays the settings page for this new class. You can select the maximum bandwidth and the limit sharing options.

To select the maximum bandwidth limit you can either:

- Select a predefined value from the drop-down list.
- Select Custom from the drop-down list and then enter your value, in kbits/second, in the box provided.

There are three possible modes for how the bandwidth class shares out bandwidth between connections:

- **Each connection is limited to the maximum rate -** No sharing is enforced, and all users of this class on all Traffic Managers will adhere to the maximum bandwidth limit set. This is useful when delivering video / audio streams to individual clients.
- **Bandwidth is shared per Traffic Manager -** The limit is shared amongst all users of the bandwidth class on each separate Traffic Manager. If you have a limit of 1 mbps and three Traffic Managers, each could deliver up to 1 mbps, giving a total of 3mbps.

- **Bandwidth is shared across all Traffic Managers -** The limit is shared among all users of the bandwidth class across all Traffic Managers. With a limit of 1 mbps and three Traffic Managers, the cluster will only deliver 1 mbps total. Note that the limit is allocated dynamically - each Traffic Manager will be capable of using the full 1 mbps if the others are idle.

Click **Update** to commit your changes.

Note: Unlike storage or memory capacity, network throughput is conventionally measured in factors of 1000 (not 1024).

Assigning a Bandwidth Class to a Virtual Server

A bandwidth management class that is assigned to a *virtual server* will limit the bandwidth that the virtual server can use when sending response data to a client.

Configuration

To assign a Bandwidth Management Class to a virtual server, click **Services**, then **Virtual Servers**. Choose the virtual server from the list of available servers, and click the **Edit** button.

Next to **Assigned Classes**, click **Edit**. Now click **Bandwidth Management**, and use the radio buttons to select one of the Bandwidth Management Classes.

Click **Update** at the bottom of the page to commit your change. The virtual server will begin using this Bandwidth Management Class immediately.

Note: if you do not have any Bandwidth Management Classes, the only option displayed will be ***none***. You can add a Bandwidth Management Class by clicking the **Create New Bandwidth Management Class** link.

Assigning a Bandwidth Class to a Pool

A bandwidth management class that is assigned to a *pool* will limit the bandwidth that the pool can use when sending request data to a node.

Configuration

To assign a Bandwidth Management Class to a pool, click **Services**, then **Pools**. Choose the pool from the list of available servers, and click the **Edit** button.

In the **Bandwidth Management** section, click **Edit**. Use the radio buttons to select one of the Bandwidth Management Classes.

Click **Update** at the bottom of the page to commit your change. The pool will begin using this Bandwidth Management Class immediately.

Note: if you do not have any Bandwidth Management Classes, the only option displayed will be ***none***. You can add a Bandwidth Management Class by clicking the **Create New Bandwidth Management Class** link.

Using TrafficScript to Select a Bandwidth Class

The bandwidth management class can also be selected per request, using either of the TrafficScript functions `request.setBandwidthClass()` and `response.setBandwidthClass()`. These functions affect the 'to-node' and 'to-client' bandwidth respectively.

A Bandwidth Management Class can be applied at any point in a TrafficScript rule, and will override the virtual server's own settings if they exist.

For example, the following TrafficScript response rule will choose the 'downloads' Bandwidth Management Class for responses which are large, or are from the "/downloads/" part of the site:

```
# This must be used as a response rule, because we
# want to determine the response content length...
$url = http.getPath();
$clen = http.getResponseHeader( "Content-Length" );

if( string.startsWith( $url, "/downloads") ||
      $clen >= 100*1024 ) {
   response.setBandwidthClass( "downloads" );
}
```

CHAPTER 17 Request Rate Shaping

This chapter explains what Request Rate Shaping is, and how to configure the Traffic Manager to rate-limit requests to your applications.

Note: Request Rate Shaping is not available on all Traffic Manager variants. If required, it can be obtained via a software or license key upgrade.

What Is Request Rate Shaping?

Individual users may dominate the use of a service, to the detriment of other users of the service. A back-end application infrastructure may have limited scalability, being easily overwhelmed when too many requests are given to it. You may wish to restrict the rate at which certain activities can occur, such as sending an email, or logging in to a service, as part of a wider security policy.

Request Rate Shaping allows you to specify limits on a wide range of events, with very fine grained control over how events are identified. You can impose per-second and per-minute rates on these events.

For example:

- You can rate-shape individual web spiders, to stop them overwhelming your web site. Each web spider, from each remote IP address, can be given maximum request rates.
- You can throttle individual SMTP connections, or groups of connections from the same client, so that each connection is limited to a maximum number of sent emails per minute.
- You may also rate-shape new SMTP connections, so that a remote client can only establish new connections at a particular rate.
- You can apply a global rate shape to the number of connections per second that are forwarded to an application.
- You can identify individual user's attempts to log in to a service, and then impede any dictionary-based login attacks by restricting each user to a limited number of attempts per minute.

Request Rate Limits are imposed using the TrafficScript `rate.use()` function:

- A virtual server accepts incoming traffic.
- A request rule is run by the virtual server.

- The request rule applies the rate shaping, using the TrafficScript `rate.use()` function.

Note: Request Rate Limits are most commonly used in TrafficScript *request* rules, to shape the rate at which requests are processed. They may be used in response rules if desired, and they can be used to restrict other events, but this is not a common requirement.

Configuring a Request Rate Shaping Class (Rate Class)

To use Request Rate Shaping, you need to add at least one Request Rate Shaping Class, or Rate class. Each Rate class defines a per-second and per-minute rate limit for events.

Rate classes are stored in the Catalog, and are invoked using the TrafficScript `rate.use()` function.

Adding a Rate Class to the Catalog

To add a new Rate class, click **Catalogs**, and then click **Rate** in the menu bar. In the box labeled **Create new Rate class**, enter a name for your new class, and then click **Create Class**.

There are two settings you can configure:

Setting	Description
max_rate_per_minute	The maximum number of requests that can take place per minute. '0' means 'no limit'.
max_rate_per_second	The maximum number of requests that can take place per second. '0' means no limit.

You can configure both per-second and per-minute limits if you wish. Both limits are applied (note that if the per-minute limit is more than 60-times the per-second limit, it has no effect).

Once you have configured the Rate class, click **Update** to commit your changes.

Using a Rate Class

Rate classes function as queues. When the TrafficScript `rate.use()` function is called, the connection is suspended and added to the queue that the rate class

manages. Connections are then released from the queue according to the per-second and per-minute limits.

Suppose a rate class named 'shape requests' is created, and it is configured with a limit of 10 events (i.e. requests) per second.

The following TrafficScript request rule will only allow 10 requests per second to be processed, no matter how rapidly new requests arrive at the system:

```
rate.use( "shape requests" );
```

As requests arrive, the request rule is executed. Provided that requests arrive at less than 10 per second, the `rate.use()` function just returns immediately and the request is processed without delay.

If more than 10 requests per second arrive, some will be queued by the `rate.use()` function. From a TrafficScript perspective, this function will block. When the request is dequeued, the `rate.use()` function then returns.

The Rate Queue

There is no limit to the size of the backlog of queued connections. For example, if 1000 requests arrived in quick succession, 990 of them would be immediately queued. Each second, 10 requests would be released from the front of the queue.

While they are queued, connections may time out or be closed by the remote client. If this happens, they are immediately discarded.

You can use the `rate.getBacklog()` function to discover how many requests are currently queued. If the backlog is too large, you may decide to return an error page to the user rather than risk their connection timing out:

```
if( rate.getBacklog( "shape requests" ) > 100 ) {
   http.redirect( "http://some.other.site/" );
} else {
   rate.use( "shape requests" );
}
```

This rule ensures that no more than 100 requests are ever queued by the rate class.

Avoiding Queuing

Alternatively, the TrafficScript function `rate.use.noqueue()` can be used to rate-shape requests in the same manner as `rate.use()`, with the key difference that requests are never queued:

- `rate.use.noQueue()` immediately returns the value 1 if the rate limit has not been exceeded. The connection rate is incremented. This is identical to `rate.use()`.
- `rate.use.noQueue()` returns the value 0 if the rate limit has been exceeded (whereas `rate.use()` would queue the request). You can then decide, using TrafficScript, what you would like to do with the request, such as returning a 503 Too Busy error.

Selective Rate Shaping

You may wish to just rate-limit particular types of request. For example, if you are using a mixture of Java extension pages and other content (such as images, static content etc.) on your Web site, you can rate-shape just the Java extension pages:

```
$url = http.getPath();
if( string.endsWith( $url, ".jsp" ) ) {
   rate.use( "shape requests" );
}
```

More Fine-Grained Rate Shaping

The examples above all apply a global limit. All requests are put into the same rate queue.

In many circumstances, you may need to apply more fine-grained rate-shape limits. For example, imagine a login page; we wish to limit how frequently each individual user can attempt to log in to just 2 attempts per minute.

With a global limit, the effect would be to restrict accesses to the login page to 2 per minute in total. We wish to impose independent per-user limits instead, so that each user is restricted to 2 logins per minute.

It would be possible to create a unique rate shaping class for each user, and select the appropriate class using TrafficScript, but there is a much better solution.

The `rate.use()` function can take an optional 'key' which identifies a new instance of the rate class. This key can be used to create multiple, independent rate classes that share the same limits, but enforce them independently for each individual key:

```
$url = http.getPath();
if( string.endsWith( $url, "login.cgi" ) ) {
   $user = http.getFormParam( "username" );
   rate.use( "login limit", $user );
```

```
}
```

The rate shaping limits are applied independently to each different value of $user. As each new user accesses the system, they are limited to 2 requests per minute, independently of all other users who share the "login limit" rate shaping class.

Rate-Shaping Web Spiders

Occasionally, a poorly written Web spider can overwhelm a Web site with requests, using excessive bandwidth and reducing the level of service that legitimate site users receive.

The following TrafficScript rule uses a rate class named 'spiders' to limit individual spiders to 1 request per second:

We identify spiders as users who do not prefix their User-Agent header with either 'Mozilla/' or 'Opera/'

We apply limits to each individual spider instance, identified by the user agent and source IP address:

```
$ua = http.getHeader( "User-Agent" );

if( ! string.startsWith( $ua, "Mozilla/" ) &&
    ! string.startsWith( $ua, "Opera/" ) ) {

            $ip = connection.getRemoteAddress();
            rate.use( "spiders", $ua.$ip );

}
```

A more sophisticated rule might only restrict users to 1 request per second if they had previously exceeded a rate of, perhaps 10 per second over a period of 5 seconds, in order to dynamically detect spiders. It could work as follows:

Maintain a list of known offenders (user agent and IP address) using the `data.set()` TrafficScript function. This list would initially be empty.

When a request is received, check the list of known offenders, and use the 1-per-second rate if the request is to be throttled.

Otherwise, use the 10-per-second rate, but before you apply the rate limit, check the backlog. If it is too large (greater than 50 perhaps), then the user has sent too many requests; insert them into the list of offenders.

Graphing Request Rate Shaping

You can use the Activity Monitor to chart the activities of your request rate shaping classes, and you can monitor the values via SNMP.

Note that only global rate limits can be graphed or monitored in this way. You cannot graph or monitor rate limits that are applied to a specific key.

CHAPTER 18 Service Level Monitoring

This chapter explains what Service Level Monitoring is, and how to configure the Traffic Manager to monitor the response times on your network.

Note: Service Level Monitoring is not available on all Traffic Manager variants. If required, it can be obtained via a software or license key upgrade.

Introducing Service Level Monitoring

Service response times are a key metric for Web services, as lower response times imply a more responsive and usable service. The response time used by Service Level Monitoring is the time in milliseconds taken from the Traffic Manager receiving a request to then sending the first byte of response data back to the client. It does not include the time taken to return the content to the client on the Internet, and generally is not affected by the speed of the remote network.

Response time includes the time taken for internal processing on the Traffic Manager (for example, running TrafficScript rules and Java Extensions), the time taken to send the request to the back-end node, and the time taken for the node to generate and return the first data in the response, and the time takes to run any response rules. It is generally independent of the speed or latency of the client connection, so gives an accurate measure of the performance of the internal systems that that application administrator can tune and optimize.

Note: if additional data, such as HTTP POST body data, needs to be read from the client before the node will return a response, the time taken to read this data is included in the response time measurement.

Using Service Level Monitoring, the Traffic Manager can measure and react to changes in response times for your nodes, by comparing response times to a conformance value. You can obtain a graph of response times and other related information, or issue alerts or log when response times are below the conformance value. The Traffic Manager also has the ability to dynamically adjust the resources available based on response times, using TrafficScript rules.

Configuring a Service Level Monitoring Class (SLM Class)

To use Service Level Monitoring, you need to add at least one Service Level Monitoring Class, or SLM class. Each SLM class contains the definition of its ideal response time and of thresholds for alerts and logging.

SLM classes are stored in the Catalog, and each can be applied to one or more virtual servers.

Adding an SLM Class to the Catalog

To add a new SLM class, click **Catalogs**, and then click **SLM** in the menu bar. In the box labeled **Create new service level monitoring class**, enter a name for your new class, and then click **Create Class**.

There are three settings you can configure:

Setting	Description
response_time	Responses that arrive within this time limit are treated as conformant. Responses that take longer than this time are non-conformant. (Time is in milliseconds).
serious_threshold	If the percentage of conformant responses drops below this level, a SERIOUS-level event will be raised, and a SERIOUS-level message written to the event log.
warning_threshold	If the percentage of conformant responses drops below this level, a WARNING-level event will be raised, and a WARNING-level message will be recorded in the event log.

For example, you may set a response time of 1000ms, an alert threshold of 40%, and a logging threshold of 70%. This would mean that:

- If fewer than 70% of requests take less than 1000ms to return from a server, a warning entry will be written in the Traffic Manager log.
- If fewer than 40% of requests take less than 1000ms, a serious-level entry will be written in the Traffic Manager event log.

The percentage of conformant requests is calculated from an average of the last 10 seconds' of requests, comparing the number of requests that fell within the **response_time** threshold to the total number of requests monitored by that SLM class.

You can define Event Handlers to call actions when warning or serious-level events are raised by an SLM class.

Once you have configured the SLM class, click **Update** to commit your changes.

Applying an SLM Class to a Virtual Server

To apply an SLM class to a virtual server, click **Services**, and then click the **Edit** button next to the appropriate virtual server.

Scroll down to **Assigned Classes**, and click **Edit**. Next, click **Service Level Monitoring** and select the SLM class which you wish to apply.

Click **Update** to apply your changes.

Once your SLM class has been associated with a virtual server, it will be used for all requests. You can, however, change the SLM class depending on the nature of the request by using TrafficScript, as follows.

Applying SLM Classes from TrafficScript

TrafficScript includes a number of functions which can be used to read current information from an SLM Class, or change the SLM Class for a connection.

These functions are documented in the **TrafficScript Reference** (available via the Online Help pages).

Note in particular `connection.setServiceLevelClass()`, which allows you to select a particular SLM Class for the request - overriding the virtual server's default - and `slm.conforming()`, which returns the current conformance percentage value.

SLM Class TrafficScript Examples

The following examples illustrate how a default SLM Class can be overridden for certain types of requests, and how to use the performance of one SLM Class to control how you manage traffic.

"FrontPage Scripts Only" Service Level Monitoring

Microsoft FrontPage™ upload scripts can take a long time to run, particularly when the client is uploading content from a slow connection. This can even lead to a pool being marked as failed, even though there is no malfunction.

To distinguish between response times for normal content, and response times for FrontPage™ binaries, we can use a TrafficScript rule which detects FrontPage™ requests (located in `_vti_bin`).

```
$path = http.getPath();

if( string.StartsWith( $path, "/vti_bin" ) ) {
    connection.setServiceLevelClass( "FrontPage" );
} else {
    connection.setServiceLevelClass( "Regular" );
}
```

The FrontPage SLM class should allow considerably higher response times than the Regular class used for all other requests managed by the virtual server.

Prioritizing Resources with Service Level Monitoring

This example tags premium customers with a 'premium' service level monitoring class, and directs them to the 'premium' pool.

Non-premium customers can share the premium pool if the premium SLM class is functioning within its tolerance, but are directed to the 'standard' pool if the premium SLM class is running too slowly.

```
if( $IsPremium ) {
  connection.setServiceLevelClass( "premium" );
  pool.use( "premium" );
} else {
  if( slm.conforming( "premium" ) == 100 ) {
    pool.use( "premium" );
  } else {
    pool.use( "standard" );
  }
}
```

Note: `pool.use()` ends the TrafficScript rule immediately and sends the request to the specified pool.

Graphing SLM Class Conformance Rates

To obtain a graph of data for a specific SLM Class, click the **Activity** button. Underneath the current activity graph, click **Change Data**.

The Traffic Manager now displays a tree of different values which may be plotted. Expand the node marked **Service Level Monitoring Classes**.

Note: You can now select whether to plot data for one particular SLM Class, or for all SLM Classes. You can also plot values for the measurement data used by the SLM Class, for example average response times for the SLM Class. A useful graph might include total connections and total non-conforming connections for all SLM Classes on the same plot, facilitating a visual comparison of the overall performance of a system.

All of the connection data and response time data can be plotted on a per node basis. This can be used to identify nodes which are under performing. A good graph to achieve this would be to plot the mean response times of each of the nodes used by the SLM Class. To select data for a node used by an SLM Class, unfold the section with the name of the SLM Class and check the nodes you are interested in.

CHAPTER 19 Content Caching

This chapter describes the Traffic Manager's Content Caching capabilities, and explains how it can be configured to reduce the load on the back-end Web-serving nodes.

Note: Content Caching is not available on all Traffic Manager variants. If required, it can be obtained via a software or license key upgrade.

Important: See also the Content Caching section of CHAPTER 20 that discusses the effect on caching when Aptimizer functionality is enabled within your virtual server.

Introduction

Most Web servers host a wide range of different types of content, which together make up the Web sites and services that are offered. For example, a typical Web site will include standard Web pages (HTML files), images (GIF, JPEG or PNG files), CSS style sheets and JavaScript source files. Depending on the service offered by the Web site, the Web site is also likely to contain some personalized, custom content that is generated by applications hosted on the Web server.

The Traffic Manager's Content Caching capability allows the Traffic Manager to identify the content that is not customized for each request and to remember ('cache') the content. The content may be 'static', such as a file on disk on the Web server, or it may have been generated by an application running on the Web server.

When another client asks for content that the Traffic Manager has cached in its internal Web cache, the Traffic Manager can return the content directly to the client without having to forward the request to a back-end Web server.

This has the effect of reducing the load on the back-end Web servers, particularly if the Traffic Manager has detected that it can cache content generated by complex applications which consume resources on the Web server machine.

Configuring Content Caching

Content Caching is performed by an HTTP virtual server. The virtual server stores common responses in a Web cache, and replies to common requests by serving the response directly from the Web cache.

Applying Content Caching to a Virtual Server

To configure Content Caching for an HTTP virtual server, click **Services**, and then click the **Edit** button next to the appropriate virtual server.

Scroll down to **Content Caching**, and click **Edit**. Enable or disable Content Caching for that virtual server using the '**webcache!enabled**' option.

Click **Update** to apply your changes.

Important: If Aptimizer is enabled for this virtual server, the Traffic Manager will always store optimized resources, such as combined images, in the cache regardless of the **webcache!enabled** setting. However, the Traffic Manager will not attempt to cache optimized HTML pages generated by Aptimizer unless this setting is enabled. If both the cache and Aptimizer are enabled, the Traffic Manager will also cache un-optimized resources that Aptimizer fetches.

Configuring Lifetimes

The Traffic Manager will automatically cache the most frequently requested content, so making optimum use of the Web cache.

For each virtual server that uses the Web cache, you can configure how long resources are cached before they become 'stale' and are discarded:

Setting	Description
webcache!time	How long (in seconds) is a resource cached for?
webcache!errorpage_time	How long (in seconds) is an error page cached for?
webcache!refresh_time	How soon before a resource expires should the Traffic Manager begin refreshing it?

The Traffic Manager caches regular HTTP responses for the period of time set in **webcache!time**. The resource is loaded into the cache the first time that a remote client requests it; then subsequent requests for the same resource are served directly from the cache until the time expires.

Note: If the resource is changed on the back-end server, it may take up to **webcache!time** seconds before the Traffic Manager notices and the cached copy is updated. The *Forcing Stale Content out of the Cache* section of CHAPTER 19 describes how content can be forced out of the cache if it has been updated.

The Traffic Manager also caches error pages (404 Not Found, 410 Gone, 501 Not Implemented, etc), but uses the **webcache!errorpage_time** parameter to determine

how long they are cached for before they are requested again from a back-end server. Typically, you would not want to cache error messages for very long; you would want to recheck the content more frequently so that the error message could be purged and replaced with the correct content.

Note: The Traffic Manager never caches pages returned as a result of a server error, such as '503 Too Busy'.

Expiring Resources

When the cached resource expires, it is removed from the cache and the memory can be reused to cache a different resource. If a resource is requested again, it will be retrieved from a back-end server and cached again.

When a resource expires, all requests for that resource will be sent to back-end servers until the complete resource is loaded into the cache again. If there are many concurrent requests for the same resource, or if the resource takes a long time to generate, this can cause a large spike of traffic to the back-end servers.

The **webcache!refresh_time** setting addresses this problem. This value controls how the Traffic Manager speculatively updates the cache when a resource is due to expire.

For example, suppose the **refresh_time** is set to 10 seconds. In the final 10 seconds before a resource is due to expire, if the Traffic Manager receives a request for the resource, it will forward the request to the back-end server rather than using the cached response. If the back-end servers provide a valid response, this response is used to update the cache and the expiry counter is reset.

The Traffic Manager will only send a maximum of one request per second for each resource to the back-end servers during this **refresh_time** period. Other requests for the resource are met using the cached response. In this manner, the cache can be refreshed before it expires, avoiding a large flood of requests to the back-end servers.

Configuring Web Cache Memory Usage

It is important to configure the memory usage of the Web cache carefully, to ensure that:

- The cache is sufficiently large to contain all variants of the most frequently requested pages.
- The cache is not too large that it causes the host machine to swap part of the running the Traffic Manager processes to disk.
- All virtual servers share the same global cache memory, and the sizes are configured as part of the **Global Settings** of the Traffic Manager system.

To configure the Web cache memory sizes, click **System**, and then click the **Global Settings** tab. Open up the **Cache Settings** section of the **Global Settings** page:

Setting	Description
webcache!size	Upper limit for the amount of memory (in Mb) that the Web cache can consume. It can also be specified as a percentage of total system memory.
webcache!max_file_num	Limits the maximum number of entries that can be placed into the cache.
webcache!max_file_size	Determines the size of the largest file that will be inserted into the cache. This can be specified as a percentage of the total cache size, or as an absolute file size.
webcache!avg_path_length	This is the estimated average length of the path (including the query string) for resources being cached. An amount of memory equal to this figure multiplied by `webcache!max_file_num` will be allocated for storing the paths for cache entries.
webcache!max_path_length	This is an upper bound on how long a path can be when stored in the cache. Its purpose is to stop one very long URL using up excessive storage space.
webcache!verbose	Enable this setting to add a header named 'X-Cache-Info' to HTTP responses to show whether the request/response is cacheable.
webcache!normalize_query	Enables normalization of the query string.
webcache!disk	Enables usage of a disk-backed cache. Refer to the Configuring Disk-Based Caching section below.
webcache!disk_dir	If disk caching is enabled, this sets the directory where the disk cache file will be stored.

Monitoring Cache Activity

Click **Activity**, and then click the **Content Cache** tab. This page will display the contents of your Web cache, and how many variants of each response are stored. The **Content Cache** page also indicates how much memory is used by your Web cache.

If your cache is full, or almost full, either the **Entries** value will be close to the **MaxEntries** value, or the **MemUsed** value will be close to the **MemMaximum** value. In this case, when new entries are inserted into the cache, the least-used entries will be discarded before they expire. You can monitor the value of Oldest to determine the time since the least-used entry in the cache was last used.

You can also click the 'View cache contents' link on the **Virtual Server > Content Cache** page. This will give a summary of cache usage (including the percentage of allocated entries and memory that is currently in use) and a selector so that you can explore the content in the cache and manually expire it if desired.

The Activity Monitor can also be used to graph the cache usage in real-time. Click **Activity** to access the **Activity Monitor** tab. Scroll down and select the 'HTTP Content Cache' report. This cache activity information is also available via SNMP.

Configuring Disk-Based Caching

Typically, the Traffic Manager's cache resides in main memory and the size you select is determined by the quantity of RAM available after accounting for the operating system, application memory, network buffers etc.

As an alternative, the Traffic Manager may be configured to use a larger cache that is stored on a local disk. This results in an effective two-tier cache because the operating system will store the most active parts of the disk-based cache in memory, using the system's kernel disk buffers.

Configuring the Cache

To use a disk-based cache, go to the **System > Global Settings** page and locate the **Cache Settings** section of the page:

1. Configure the size of your cache, in Mb or Gb. This size might be several times larger than the amount of memory in the server.

2. Enable disk-backed caching with the **webcache!disk** setting.

3. Select a location for the disk cache. This location must be a local directory with sufficient free space, and you are strongly recommended to select a location on

fast SSD (Solid State Drive) storage, rather than on a slower local HDD (Hard Disk Drive).

Click **Update** to save your settings. To fully apply disk-based caching, you will need to restart your Traffic Manager software via the **System > Traffic Managers** page.

The disk cache is not persistent, and is cleared and reset when the Traffic Manager software is restarted. You cannot share disk caches between multiple Traffic Manager systems.

SSD and HDD Disk Caches

The disk cache is stored in a single flat file created in the selected location. All reads and writes to the cache can potentially result in a disk access, although the host operating system will cache the most commonly accessed parts of the file and perform writes in the background.

You are very strongly recommended to use a local SSD drive for the cache files because reads, writes and seeks on these drives are very fast. Reads, writes and seeks on a HDD are much slower – a typical mechanical HDD is limited to several hundred disk operations per second, which then limits the number of requests per second the Traffic Manager can process via the cache.

Caching Policy

Not all Web content can be cached. Information in the HTTP request and the HTTP response drives the Traffic Manager's decisions as to whether or not a request should be served from the Web cache, and whether or not a response should be cached.

Requests

Only HTTP GET and HEAD requests are cacheable. All other methods are not cacheable.

The `Cache-Control` header in an HTTP request can force the Traffic Manager to ignore the web cache and to contact a back-end node instead.

Requests that use HTTP basic-auth are uncacheable.

Responses

The `Cache-Control` header in an HTTP response can indicate that an HTTP response should never be placed in the web cache.

The header can also use the `max-age` value to specify how long the cached object can be cached for. This may cause a response to be cached for less than the configured **webcache!time** parameter.

HTTP responses can use the `Expires` header to control how long to cache the response for. Note that using the `Expires` header is less efficient than using the `max-age` value in the `Cache-Control` response header.

The `Vary` HTTP response header controls how variants of a resource are cached, and which variant is served from the cache in response to a new request.

If a Web application wishes to prevent the Traffic Manager from caching a response, it should add a '`Cache-Control: no-cache`' header to the response.

You can use the global setting **webcache!verbose** if you wish to debug your cache behavior. This setting is found in the **Cache Settings** section of the **System, Global Settings** page. If you enable this setting, the Traffic Manager will add a header named '`X-Cache-Info`' to the HTTP response to indicate how the cache policy has taken effect.

For further details about HTTP cache policies, please refer to RFC 2616.

Variants

A URL may have multiple variants if:

- The response depends on additional request parameters, such as the Accept-Encoding or Accept-Language headers. The origin web server will indicate that this is the case by including a 'Vary' header in the response:

```
Vary: Accept-Language, Accept-Encoding
```

- The TrafficScript '`http.cache.setkey()`' function has been used to indicate the response content depends on other computed parameters.

Byte Ranges

Byte-ranges are used by some client software to request a portion of a document. For example, a PDF reader may download the first part of a PDF file (which contains an index of pages), and then issue a byte-range request to retrieve the pages the user wishes to read.

The Traffic Manager treats byte-ranges as a variant of a URL (see the *Variants* section of CHAPTER 19). Byte range responses are cached, and if another request for the same byte-range is received, the response will be served directly from the cache.

If the Traffic Manager caches the entire response for a URL, the Traffic Manager will then serve all byte-range requests from that response.

ETags

ETags (entity tags) are used by Web servers to uniquely identify each version of a response. The Traffic Manager caches and honors ETags and will respond with a 304 Not Modified response if the current cached version of a resource has the same ETag as the version cached by the remote client.

Controlling Content Caching Using TrafficScript

HTTP Request Processing

When the Traffic Manager receives a new HTTP request, it must determine whether it can serve a response directly from the Web cache, or whether it should forward the request on to a back-end server. The Traffic Manager executes all TrafficScript request rules on the request before it inspects the Web cache. Consequently, a TrafficScript request rule can modify a request and influence the Traffic Manager's decision.

For example, a TrafficScript request rule could add a '`Cache-Control: no-cache`' header to a request to prevent the Traffic Manager from serving the response from the local Web cache.

HTTP Response Processing

Similarly, when the Traffic Manager receives an HTTP response from a back-end server node, it must decide whether or not to insert the response into the Web cache for future use. The Traffic Manager executes all TrafficScript response rules before making this decision, so a TrafficScript response rule can modify a response and influence this decision.

A TrafficScript response rule could add a '`Cache-Control: no-cache`' header to the response to prevent it from being cached, or a '`Cache-Control: max-age=60`' header to indicate that the response should be cached for 60 seconds.

Note that if a response is served from the Web cache, the TrafficScript response rules are not executed.

TrafficScript Cache Control Functions

Several cache control functions are available to facilitate the control of the Traffic Manager's caching behavior. In most cases, these functions eliminate the need to manipulate headers in the HTTP requests and responses.

Function	Description
http.cache.disable()	Invoking http.cache.disable() in a response rule prevents the Traffic Manager from caching the response.
http.cache.enable()	Invoking http.cache.enable() in a response rule reverts the effect of a previous call to http.cache.disable(). It causes the Traffic Manager's default caching logic to take effect. Note that it is not possible to force the Traffic Manager to cache a response that it deems to be uncacheable.
http.cache.setkey()	The http.cache.setkey() function is used to differentiate between different versions of the same request, in much the same way that the `Vary` response header functions. It is used in request rules, but may also be used in response rules. It is more flexible than the RFC2616 Vary support, because it lets you partition requests on any calculated value – for example, different content based on whether the source address is internal or external, or whether the client's `User-Agent` header indicates an IE or Gecko-based browser.
http.cache.exists()	The http.cache.exists() function can be used in a request rule to check if a matching response exists at that instant in the cache. If the rule were to terminate without modifying any of the request parameters, the response would be served from the cache (unless it had expired in the intervening time).

Function	Description

http.cache.respondIfCached()	This function can be called in a request rule. If a matching response exists in the Web cache, the rule terminates immediately and the response is sent back to the remote client. No further request (or response) rules are run.
	
	If a matching response does not exist, the function returns and rule processing continues.
	
	This can be used to support logic such as rate shaping, whereby you wish to rate-shape requests to back-end nodes, but do not need to apply a rate limit if you can serve the response directly from the Traffic Manager cache:
	
	`http.cache.respondIfCached();`
	`rate.use( "my rate limit" );`

For more information and code samples, see the *Brocade Virtual Traffic Manager: TrafficScript Guide* available from the Brocade Web site (http://www.brocade.com).

Forcing Stale Content out of the Cache

If you update part of the content for your load-balanced Web site, the Traffic Manager will only pick up the new content when the existing cached versions expire.

You can manually force the Traffic Manager to expire content from the cache, so that it will retrieve the updated versions next time a client requests the content, or you can do so programmatically.

Manual Removal of Cached Content

To manually remove cached content, perform the following steps:

1. Go to the **Activity > Content Cache** page in the Traffic Manager Administration Interface; this page will display a selection of the files that are currently cached by your Traffic Manager cluster.
2. If necessary, use the '**Display Cache Items**' selector to locate the items you want to expire, specifying host header matches and path matches to narrow the selection down.
3. Select the entries you wish to expire and press the '**Clear Selected**' button.

This will immediately invalidate the cached copies of these resources, and the Traffic Manager will retrieve new versions the next time a client requests them.

Programmatic Removal of Cached Content

The Traffic Manager Control API includes SOAP methods to inspect and invalidate entries in the cache; these methods are located in the System.Cache interface.

A content provisioning system that uploads new content to the Web servers could potentially also invoke a script that used the Traffic Manager Control API to also invalidate any updated items from the content cache.

For more information, see the *Brocade Virtual Traffic Manager: Control API Guide*.

CHAPTER 20 Aptimizing Your Web Content

Introduction

Faster-loading Web pages provide users with a more satisfying experience because they reduce waiting time for a Web page to render and be usable. Studies show that with an average Web page load time of six seconds, only about one second is spent inside the application and Web server, with the user waiting another four to five seconds for the client browser to render the Web page. Reducing Web page load time is vital to improving the overall user experience.

Aptimizer automatically optimizes Web page markup and elements so they load faster for end users. The technology is present within your Traffic Manager infrastructure and no changes are required to your back-end applications.

Modes of Operation

Aptimizer is an optional Traffic Manager feature that is enabled through a specially configured license key. Two levels of functionality are available, depending on your license key: Express and Enterprise.

Aptimizer Express applies a predefined level of optimization to your Web content, designed to best match a wide range of applications. In this mode, a virtual server with Aptimizer enabled compresses the HTML contained in Web pages it processes. It also performs the following optimizations on the page resources as the client fetches them:

- Minification and compression of JavaScript files.
- Minification and compression of style sheets.
- Background images are inlined or versioned.
- Web fonts are versioned.
- Resampling of image content.
- Compression of all resources.

See "Understanding Optimization Techniques" for more information on each of these items.

Important: In Express mode, Aptimizer cannot be used to optimize sites requiring authentication. See "Authentication" for more details.

Note: Aptimizer-specific TrafficScript functions are limited to operating with their default parameters in Express mode, and cannot be overridden. See the *Brocade Virtual Traffic Manager: TrafficScript Guide* for more information.

An Enterprise license permits Aptimizer to inspect and modify the content of HTML pages, enabling it to perform a full range of optimization techniques. Aptimizer Enterprise also provides full control over the individual optimization parameters that are applied to your Web infrastructure. You can create and apply specific optimization profiles to individual application scopes handled by your virtual servers.

This chapter provides a complete description of the features and capabilities of Aptimizer when running in Enterprise mode. Some elements do not apply to Express mode.

Configuring Aptimizer for Your Services

Aptimizer can be enabled as a property of a virtual server, through the **Aptimizer settings** link on the **Virtual Servers > Edit** page of the Admin UI.

It is possible to fully enable or disable Aptimizer functionality for a virtual server using the **aptimizer!enabled** configuration key in the Basic Settings section.

Note: If you are running Aptimizer Express, the UI controls discussed in the remainder of this section are not applicable and should be considered for reference only. Contact your support provider for more information.

A virtual server manages traffic for a specified port and protocol. Requests that it handles might be for a particular application, based upon the URL requested; for example, http://www.mywebsite.com/supportsystem. This hostname and path combination can be used to create a connection between the application being requested (through this virtual server) and the optimization profile that should be used when it is identified.

In the Aptimizer Profiles section, you can define the mappings between various acceleration *Profiles* you wish to use and the *Scope* within which they should apply. Both of these concepts are **Catalog** objects and must be created before being used on this page. Each of these concepts is explained below.

You can define and map several Scopes to a single Profile, depending on your requirements, though a scope can only be used once within a single virtual server.

The Aptimizer Wizard

As with other wizards, the Traffic Manager provides an automated process to enable you to quickly set up Web content optimization on your Web services: the **Aptimize a Web application** wizard.

You must have previously created at least one virtual server in order to do this; the wizard cannot run without one. It is recommended that you first follow the **Manage a new service** wizard in order to create a suitable virtual server upon which to run the wizard.

To run the wizard, select **Aptimize a Web application** from the "Wizards" drop-down menu in the navigation bar.

This process has three main steps, in which you will select:

1) The virtual server upon which you are applying optimization

2) The Application scope

3) The Aptimizer profile to be applied for that scope

Application Scopes

An *Application Scope* object is used to define a collection of URLs that match a specific Web site or application hosted by a virtual server. You can find these on the **Catalogs > Aptimizer > Application Scopes catalog** page.

When a request is received through a virtual server, a match is sought to the most-specific URL within the Scope mappings defined in the **Aptimizer** settings page. If the Traffic Manager finds a match, it applies the appropriate optimization Profile defined in the mapping.

Several hostnames can be included in one Scope record. However, all these names should be aliases for the same host, and it should be possible to fetch the same resources using any of them. If an alternate hostname is used to serve content from a Content Delivery Network, then that CDN must proxy the origin host so that content passes through the same Application Scope again within the Traffic Manager.

To match 'any' hostname, create a new scope record and leave the `hostnames` list blank. Alternatively, where multiple applications are hosted through a particular virtual server, differentiated by the URL in the request, set up a new Scope record for each combination of hostname and root path involved.

Resources served through the virtual server will only be considered for optimization if they match both the hostname declared in the attached scope and the port on which the virtual server is running. The hostname match will be performed using

the hostname(s) specified in the scope, or using the hostname of the page request if the scope hostname setting is blank. All requests for dependent resources will use the primary hostname (first in the list) if explicitly set.

To optimize resources hosted on a different host, such as an external media host, use an Aptimizer Profile that includes a URL rule with the **Applies to externally hosted resources** option enabled. Please refer to the Acceleration Settings section below for more details.

Applying an Application Scope that specifies a hostname to a virtual server that does not host an application by that name may cause requests to fail with a 502 status - if you see this HTTP status, check that the Application Scope hostnames are logically grouped by the application they represent.

Note: You should ensure that multiple scopes do not provide conflicting URL matches.

When creating a new scope entry, the `root` config key can be utilized to further identify the root path to the application in scope.

A built-in scope is automatically provided, entitled "**Any hostname or path**", that can be used where no specific hostname and/or path identification is required within the Web application. The hostname field is blank, and the root field is set to '/' accordingly.

Aptimizer Profiles

The Aptimizer Profiles catalog contains a list of profiles that can be applied to websites and Web applications to enable automatic content optimization. You can find these on the **Catalogs > Aptimizer > Aptimizer Profiles catalog** page.

Built-in and Custom Profiles

The Traffic Manager provides a number of **built-in** profiles that provide pre-defined optimization settings for your Web applications. These profiles are not modifiable or deletable, and are designed to offer a choice of optimization settings applicable to most situations. Choose from:

Name	Description
Express	This profile provides basic optimization settings and techniques designed for a wide range of web sites. These techniques include shrinking CSS, inlining CSS background images, minifying javascript, shrinking images, and advanced caching.

Name	Description
SharePoint 2007	This profile provides optimization settings tuned for acceleration and compatibility with Microsoft SharePoint 2007 Internet and intranet sites. It includes resource URL versioning, durable caching, CSS StyleSheet inlining, CSS minification, background image inlining, JavaScript minification, compression, image shrinking and metadata removal, and advanced JavaScript compatibility.
SharePoint 2010	This profile provides optimization settings tuned for acceleration and compatibility with Microsoft SharePoint 2010 Internet and intranet sites. It includes resource URL versioning, durable caching, CSS StyleSheet inlining, CSS minification, background image inlining, JavaScript minification, compression, image shrinking and metadata removal, advanced JavaScript compatibility and SharePoint ribbon optimization.

Name	Description
SharePoint 2013	This profile provides optimization settings tuned for acceleration and compatibility with Microsoft SharePoint 2013 Internet and intranet sites. It includes support for a SharePoint feature called Minimal Download Strategy (MDS). This is designed to reduce the white-screen effect when navigating between pages and to reduce the data transferred. As your browser navigates from page to page, the HTTP page responses contain only fragments of HTML that differ from the previous page. Without MDS enabled, each page response contains the entire page content. The MDS feature is enabled by default, but it is disabled after the site *look and feel* is customized. Differences to the SharePoint 2010 profile are: ▪ Aptimizer supports and optimizes MDS fragments. ▪ SharePoint 2013 loads many individual scripts. Aptimizer optimizes this behavior by delivering script content through preconfigured script sets. For example, when SharePoint requests "/_layouts/15/init.js", Aptimizer delivers a script set containing that and other scripts normally loaded at the same time. When SharePoint needs one of these scripts, it is already in the browser cache as part of the Aptimizer script set and can be used immediately.
SharePoint 2013 Custom Website	The SharePoint 2013 user interface can be heavily customized, and might not use all the default scripts, styles, and images that a default SharePoint 2013 site uses. This profile is an adapted version of the standard "SharePoint 2013" profile, designed for customized Web sites that do not require the default SharePoint scripts and resource sets to be downloaded by browsers. For default SharePoint sites, use instead the "SharePoint 2013" profile.

Name	Description
Other Web application	This profile provides basic optimization settings appropriate for safely accelerating most Web applications. It includes CSS StyleSheet inlining, CSS minification, background image inlining, JavaScript minification, compression, and image shrinking and metadata removal. You can use this profile as a template for creating new profiles customized to the optimization requirements of your Web applications.
Basic (deprecated)	This profile is designed to enable a useful number of basic optimization techniques that work well on a wide range of Web sites. These techniques include shrinking CSS, in-lining CSS background images, *minifying* JavaScript, shrinking images, URL versioning and advanced caching.
Advanced (deprecated)	This profile enables more advanced optimization techniques in addition to those offered by the Basic profile, including combination of CSS style sheets and combination of HTML images into image sprites. These techniques will dramatically reduce page load time, but may require customized rules for compatibility with some sites. In this case, you would clone the profile in order to create a custom Aptimizer profile (discussed below).

If these profiles cannot offer you the desired level of optimization across all elements of your Web site, you can opt to create a new **custom** Aptimizer Profile that fulfills your specific requirements.

New custom profiles can be created in the *Create a new Aptimizer Profile* section using one of the built-in profiles as a template. Select the desired template profile from the drop-down list and click the **Create Aptimizer Profile** button. Built-in profiles can also be duplicated to create new custom profiles using the **Save As** option on the *Edit profile* page.

Custom profiles provide a range of additional acceleration options, as discussed in the Understanding Custom Acceleration Profiles section below.

Optimization Modes and Page Information

Aptimizer can run in one of three modes, defined using the `aptimizer_mode` configuration key:

- **Off:** Acceleration is disabled, but requests for Aptimizer resources are served.

- **Stealth:** Acceleration is disabled, except where enabled using the query string command:

```
?aptimizer=on
```

- **On**: Acceleration is enabled for all requests.

You can elect to show request information for each Web page that is passed through an Aptimizer Profile using the `show_info_bar` configuration setting on the Edit Profile page. If this is set to **Yes** and acceleration is currently active, all pages in this session will show a status bar which includes controls to provide information specific for that request.

Background Optimization

Optimizing complex resources, such as large image sets or CSS files with multiple nested imports, can be a time-consuming process. Although Aptimizer will do this only once for a given resource, it might result in the client that first requires this optimized resource having to wait for it to be completed.

This section allows you to configure Aptimizer to complete optimizations in the background, serving the original un-optimized server content to all clients that request it until the optimized data is ready.

Note: Resources, such as combined image sets, will be automatically cached after they are optimized. Web pages will be cached only if the virtual server's `webcache!enabled` setting is **Yes**. However, if all the resources on the page have already been optimized, the fully optimized page can be delivered quickly.

The Traffic Manager offers two profile settings for determining whether and how to optimize resources in the background:

Setting	Description
background_after	Specifies the number of milliseconds after which the optimization will be put into the background and the original server content will be served to the client. When set to 0, Aptimizer will always wait for the optimization to complete before sending a response to the client. **Note**: Some resources, such as JavaScript files, must always be served optimized. As such, they cannot be optimized in the background using this setting.

Setting	Description
background_on_additional_resource	When a page, or a resource such as a CSS document with nested imports, requires Aptimizer to fetch additional resources to complete the optimization process, do the optimization in the background.

Measuring Aptimizer Changes

This section explains how to test Aptimizer and measure Web page performance using a variety of tools. These measurements can be used to determine how effectively Aptimizer is accelerating your Web service.

Checking That Aptimizer Is Active

To check that your Web content is being accelerated, use your browser's "View HTML Source" option after loading a Web page. Search for the following comment:

```
<!-- Optimized for speed - Stingray Aptimizer -->
```

If you do not see this comment, see "Configuring Aptimizer for Your Services" on page 288.

Using Stealth Mode to Test Aptimizer

If you do not have a development or test environment available you can still test Aptimizer by running the accelerator in Stealth Mode.

Note: Stealth Mode is not available with Aptimizer Express.

In Stealth Mode, Aptimizer will not apply any optimization to Web pages, unless commands are passed via the query string. End users will not see any change in the way Web pages behave or appear.

When running Aptimizer in Stealth Mode all incoming requests for pages and objects will be served in their original form. If the request contains the parameter `aptimizer=on` in its query string, Aptimizer will send a cookie that represents an explicit instruction to accelerate all requests for this browser session.

When Aptimizer sees the enabling cookie on the incoming requests it will intercept and accelerate them according to the rules and settings defined in the Aptimizer profile applicable to that scope.

These are examples of valid requests:

```
www.example.com?aptimizer=on
www.example.com/destinations?aptimizer=on
```

You configure Aptimizer to run in Stealth Mode by changing the **aptimizer_mode** configuration key in the relevant profile.

Measuring Web Page Speed

To find exactly what has changed since enabling Aptimizer you will need before and after Web page load time measurements.

An initial measurement should be used as a baseline to which further measurements will be compared. Ideally the baseline should be documented while the Web site is not yet running with Aptimizer enabled, or with Aptimizer running in Stealth Mode.

At each stage a new set of measurements can be taken and used for comparison with the baseline. The stages could be any milestone in the Aptimizer deployment, for instance:

- Initial deployment with default settings
- Changes to rule options
- Changes to CSS settings
- Changes to caching behavior
- Changes to default image sets

Tools

There are many tools available to verify Web page load times and provide feedback on Web page structure. Some can be used in a very granular fashion, allowing testing on individual Web pages. Others are used for collecting metrics that can be correlated to the Aptimizer-enabled implementation.

Page Speed and Structure

Many tools can test and provide information about your Web pages. Some are available as online services while some are available as an add-in for Web browsers such as Firefox or Chrome. Some typical examples follow, however your exact requirements may vary:

- WEBPAGETEST (www.webpagetest.org) is a free web site service that emulates browser behavior and retrieves your web page for analysis, presenting results in a cascade-style chart showing how a browser progresses through the page when rendering it.

  - A simple process for ensuring that the site is as fast as possible is to work your way down the stepped list of objects on the chart, determining the most appropriate acceleration to apply for each object. For example, the start of the chart might show two redirects and then the page loading. The ideal action for this would be to rework the site such that the redirect is not used. Or you may be looking further down the chart at a script that loads on the page.

- YSlow (http://developer.yahoo.com/yslow/help) is a browser add-in developed by the Yahoo! Developer Network initiative. The software uses built-in rules to grade a web page across a range of performance indicators, from A (best) through F (worst) and provide suggestions to improve the web page score. It runs on Firefox and Chrome browsers.

- Google Page Speed (http://code.google.com/speed/page-speed) is another browser add-in also available for Firefox and Chrome browsers. The software grades web pages on a Page Speed Score from 0 (worst) to 100 (best). Google Page Speed also provides suggestions on how to improve the web page score.

- Brocade makes the Aptimize Site Analyzer tool available to customers and partners. The tool runs on a Windows-based PC to load and analyze a web page, providing a cascade-style chart and timings all in a single tool. It also generates a PDF file with summary findings.

Other Metrics

Some metrics require a longer data collection period to be useful. This means collection must start even before Aptimizer is enabled.

- Google Analytics (www.google.com/analytics) is a free tool that can be used to collect metrics such as page load time, number of page views, page views per visits and bounce rate. Google Analytics requires a small script to be added to all pages in your web site. Results are visible a couple of days after installing the script, but you need to collect data for some time to allow for effective comparisons.

- Google Webmasters (www.google.com/webmasters) is another free tool that can be used to manage how Google sees your web site. As a bonus it provides a chart showing the average global web page load time for your entire web site, over a rolling window of six months. Google Webmasters requires site ownership authentication, which can be accomplished by a small DNS change, a configuration file added to your web server, or a meta tag added to your default web page.

Understanding Custom Acceleration Profiles

Note: This section is not applicable to Aptimizer Express.

Typically, the built-in Aptimizer acceleration profiles provide good general-purpose optimization for your Web applications. However, there might be times when you need to define *custom* acceleration profiles in order to provide a more fine-grained control over optimization of specific elements of your Web site or application. You can find an overview of profiles in the Aptimizer Profiles section above.

The Edit page for a custom acceleration profile includes an additional control referred to as the **Acceleration Settings** dialog.

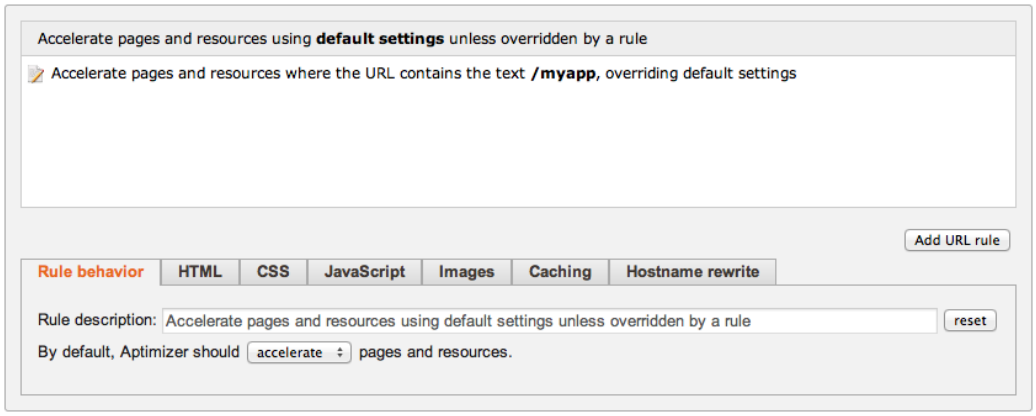

Accelerate pages and resources using default settings unless overridden by a rule
Accelerate pages and resources where the URL contains the text /myapp, overriding default settings
Add URL rule
Rule behavior
HTML
CSS
JavaScript
Images
Caching
Hostname rewrite
Rule description: Accelerate pages and resources using default settings unless overridden by a rule
reset
By default, Aptimizer should accelerate pages and resources.

Fig 54. The custom profile accelerations setting dialog

This contains controls and rules to handle optimization of different content types within your Web application. It is split into two main parts:

- The top part shows the hierarchical list of rules that are applied to your web content, usually based on URL location or browser type.
- The set of tabs below this represent the properties that apply to the current rule. As you highlight a particular rule, its properties are displayed in the tab set below it.

A **default settings** rule is provided for each Acceleration Profile that cannot be removed. This allows you to define the default behavior of the profile and configure the fall-back values for each of the property tabs. New rules can be added beneath this to further define optimization behavior and to override the default properties based on a specific URL (click **Add URL rule**). These newly added rules can be ordered according to priority or desired effect by clicking on the up or down arrows to the right of each rule bar. They can be removed altogether by clicking the delete button to the right.

In the rule properties section of the dialog, each displayed tab contains settings relevant to content that constitutes a typical Web application, such as CSS, JavaScript and images. The first tab, Rule Behavior, contains the general description of this particular rule and governs its overall behavior within the scope of whether to accelerate or exclude the content defined by the rule.

Profile Default Rules

In order to provide compatibility for mobile browsers of limited capability, both the Advanced and Basic profiles include settings to selectively disable some Aptimizer features for these browsers.

Specifically, the following browsers have *image combination*, *stylesheet combination* and *CSS image inlining* disabled by default:

- IE mobile 7
- Opera mini
- Opera mobile
- Blackberry

Additionally, certain optimizations are disabled for search crawler agents in order to allow cached versions of pages to be generic for all browsers. *Image combination* and *CSS image inlining* is disabled for the following search crawler user agents:

- bingbot
- googlebot

Note: These profile default rules are not visible in the custom rules view of the Aptimizer configuration.

Overriding Default Behavior

Each new URL rule added to the list has its own set of properties that can override the default behavior when a match is made. Each property tab contains an **Override** checkbox that enables or disables this mechanism for that particular property tab. Should you wish to provide an alternate configuration setting for a particular rule, click the Override checkbox to enable the settings for that tab. By default, all settings are disabled/greyed-out until the override checkbox is selected.

Rule Chaining

The default behavior in any given situation is for each request to pass through the list of URL rules and stop when a match is found. This *stop-when-matched* behavior can be overridden by selecting the **Allow settings for this rule to chain to other rules** checkbox within a rule.

If enabled, a matching request will not stop at that rule. Instead it will continue on to find a second rule match. Note, however, that this setting only remains in effect until a further match is found. At that point, if the second matching rule does not also have chaining enabled, the request will not attempt to match any further rules. If you wished to have all rules tested throughout your profile, regardless of how many matches occur, you would need to enable chaining on each of them.

Acceleration Settings

Each group of related optimization settings are shown on a separate tab. These are listed below.

Rule Behavior Tab

Setting	Description
Rule Description	A textual description of the current rule, automatically assembled from the basic rule behavior settings contained in this tab. For example, changes to the URL test string or rule mode will be automatically reflected in the descriptive text in this field. It is possible to override this text with your own description, however no further automatic updates will be made. To reset the description back to the default auto-generated string, click the **reset** button.
Rule mode	Aptimizer can be set to **accelerate** or **exclude** all pages and resources identified by this rule. Accelerate mode is the default, and performs the desired optimization defined within the rule properties on a matching request. The alternative is to set the rule to exclude optimization for pages or resources that match the supplied URL test. A third mode, **strip**, strips out resource references where a match is encountered.

Setting	Description
URL test	These rules only apply to requests that match the specified text. The text is compared with the full URL including the hostname if the **Match external resources** box is checked, otherwise it is compared only with the URL path. You can specify a substring that must be contained in the URL, or a wildcard or regular expression that must match the entire URL.
Applies to externally hosted resources	This setting allows you match against resources held external to the Web application. If enabled, the rule will only match requests for hosts that are not included in the Application Scope, and the string contained in the **URL test** field will be matched against the hostname and path contained in the request. For example, you may wish to exclude media files on your Web page that are included from an external media hosting Web site. If this is the case, you can select this option and specify a wild-carded string in the **URL test** field that includes the name of the hosting site in addition to the URL path of the media files you are interested in.
Rule chaining	Allow settings for this rule to chain to other rules. Please refer to the Rule Chaining section above for more details.

CSS Tab

Setting	Description
CSS and Stylesheet optimization	**None:** Style sheets will remain in their original form wherever they appear. **Optimize CSS Content:** Style sheet blocks in the page and in external style sheets will have all redundant whitespace and comments removed. **…and combine consecutive stylesheets**: Style sheets that appear as consecutive references (i.e. without inline style blocks or excluded style sheet links between) will be combined and each combined reference inserted to the location where the first style sheet in a set appears in the document. **…and combine all stylesheets**: All style sheets will be combined, no matter where they appear in the page. The new combined reference will be inserted at the top of the head section of the document.
CSS background image optimization	The maximum allowable size (in bytes) of base64 encoded data to be inlined using data:uri notation or a MHTML reference. Images that exceed this limit will remain as URLs.

JavaScript Tab

Setting	Description
JavaScript optimization	**Enabled:** Script files will have all redundant whitespace and comments removed. **Disabled:** Script files will remain in their original form.

Images Tab

Setting	Description
Combine images	**Enabled:** Images that appear as <img> tags in the HTML are combined using one of the following techniques: ▪ **CSS image spriting**: Images are combined into an image sprite according to the image format. ▪ **Image streaming**: Images are combined into one or more image streams. **Disabled:** Images that appear as <img> tags in the HTML are not combined.
Shrink images	Images are shrunk where possible, removing unnecessary information such as metadata, redundant color information and in the case of JPEG images, reducing the quality to reduce the size of the files. If an image appears on a Web page in BMP format, the Traffic Manager automatically converts it to JPEG format to reduce its size. You can prevent this by using a rule to exclude the image from being processed.
Resample quality	The percentage quality level to resample *lossy* images to (0-100). If retaining the quality of your images is paramount, set this value to 100. The default is 85.
Convert images	Allow images to be optimized to a different format than the original. The format can be chosen explicitly, or set to "Optimal", which chooses the best format automatically for each image.
Resize images	Resize images to a specified width and height in pixels.

Caching Tab

Setting	Description
Honor resource Cache-Control header	**Auto:** Aptimizer determines whether to cache a resource based on its content type. Static resources, such as style sheets and images are always cached, whereas dynamic resources, such as JSON-formatted API responses, obey the response's *Cache-Control* directives. **Always:** Aptimizer obeys all *Cache-Control* directives in the response. **Never:** Aptimizer caches all content it optimizes.
Version resource URLs	**Enabled:** URLs at this location are versioned by rewriting the URL to contain a hash of the contents of the file. The cache expiry time for versioned URLs is 1 year. **Disabled:** URLs are unaltered and clients cache the item for the full **Cache on browser** interval before rechecking. See "Resource Naming and URL Versioning" on page 309 for more information.
Mark resources as cacheable by intermediary proxies	**Enabled:** Resources that are cacheable at this location will be marked with the *Cache-Control* public header that allows them to be cached by intermediary proxy servers. **Disabled:** Resources that are cacheable will be marked with the *Cache-Control* private header to prevent them from being cached by intermediary proxy servers.
Optimize resources requested directly by their original URL and store them in the cache	**Enabled:** If a resource such as an image or CSS stylesheet is requested by its original URL, enabling this option will cause Aptimizer to optimize and cache the resource before returning it to the client. This can be useful if, for example, your site contains JavaScript that fetches images or other resources from your server after the page has loaded. **Disabled:** Directly requested resources are returned un-optimized and will not be cached by the Traffic Manager.

Setting	Description
Cache on browser	The duration, in seconds, resources will be cached on the client's browser before expiring. This is used as a sliding expiry to set the *Expires* header, meaning the client will cache the file for the specified number of seconds from when it receives it. This setting applies only to resources that are optimized and requested by their original URL. **Note**: This excludes *combined resource sets* or resources utilizing *URL auto versioning*. In these cases the URL of the resource (or resource set) is generated from a hash of the contents, so modifying the Expires header has the effect of altering the resource URL. This means that the resource would have to be re-requested on the strength of the changed URL.
Background recheck	The time, in seconds, after which a resource that is in the server cache will be marked for a recheck. When a resource that is marked this way is requested, the existing version in cache will be returned immediately, while a background task refreshes the cached version if it has been modified.

Hostname Rewrite Tab

Setting	Description
URL hostname rewriting	Relocates resources from the current domain onto the specified host or URL root by rewriting all matching URLs. This can be specified as a single hostname (for example, cdn.example.com) or a full root URL with optional scheme and path components (for example, https://cdn.example.com/images).

Understanding Optimization Techniques

This section discusses various optimization principles employed by Aptimizer. The first part relates to the concept of *Web page speed rules* and how Aptimizer applies

those rules to optimize your Web content. We then discuss a number of techniques Aptimizer uses to apply optimization to Web applications.

Web Page Speed Rules

These broadly fall into four categories, each of which can be applied to your Web application in order to provide a better response to the client browser:

- Reduce the number of objects being loaded by Web page
- Reduce the size of everything sent to and from the server
- Cache as much as possible to speed up repeat views
- Reduce the time it takes for the server to respond to a request

The sections that follow describe each in more detail.

Reduce the Number of Objects Being Loaded by Web Pages

For each element on your Web page, the browser will issue a HTTP request to the host where the resource is stored. Browsers limit the number of simultaneous connections that can be established to each server; therefore fewer Web page elements mean a faster Web page load. It is this time optimization step that will accelerate Web page loading the most.

Aptimizer automatically reduces the number of objects by combining files when appropriate, creating **combined resource sets** by resource type. Specific information regarding the most common types is shown below:

Image In-Lining and Combination

Images are important elements in every Web page. They can be separated into CSS background and HTML images, with different approaches for each type:

- **CSS background images**

  CSS background images are referenced from the background property in a custom style sheet or directly in a style property within a Web page. To reduce the number of requests for these resources, Aptimizer will convert them to base64 data and embed this in the style sheet file or HTML.

  The exact acceleration behavior differs depending upon the browser:

  - For Internet Explorer (version 6 and 7) the base64 data is served in a separate file, with the style sheet background URL referencing this file using MHTML notation;
  - For all other browsers, the base64 data is embedded directly into the CSS using the *data:uri* notation.

- **HTML images**

  HTML images are images referenced from an `<img>` tag in a page. To accelerate these images, Aptimizer can create a combined CSS sprite that contains all images of a particular format.

  Aptimizer then attaches some CSS style information to the `<img>` tag to reference this sprite as a background image, and sets the *source attribute* to reference a transparent GIF.

JavaScript Minification

Reducing the amount of superfluous content in JavaScript files run from within the HTML can help to improve overall page load time. Content such as redundant whitespace and comments, although useful from a software development perspective, serve no functional purpose in the operation of the page so are removed in order to reduce the file size.

Style Sheet Combination

Reducing the style sheet request count is an important step in accelerating the time a browser starts rendering a Web page. Combining style sheets is generally a safe and effective way of accelerating page load time.

Accelerating Third-Party Domain Content

By default, Aptimizer will apply Web page acceleration rules to elements served from the same Web subdomain from which the page is served. Consider the example of an image URL in the `<img>` tag for the Web site `www.example.com`:

URL	Result
`/images/sample1.jpg`	Aptimizer accelerated
`www.example.com/images/sample1.jpg`	Aptimizer accelerated
`images.example.com/sample1.jpg`	not accelerated
`images.otherexample.com/sample1.jpg`	not accelerated

It is possible to modify this configuration to include other Web subdomains, even third party domains, effectively accelerating partners' content required to display your Web pages.

Aptimizer can rewrite these resource references to either:

- Combine the object into a CSS sprite set.
- Apply versioning (see the "Resource Naming and URL Versioning" section on page 309) to the original URL, which will be served from the domain host.

A technique to increase the number of concurrent resource downloads is “domain sharding”, where some of these resources are downloaded from different Web domains or subdomains. This approach provides more benefits than domain sharding, because every distinct Web domain name used on a Web page requires additional DNS resolution time and TCP connection establishment time. Also, the chance of some content being unavailable is greatly reduced, since this content is now being served from your Web server.

Reduce the Size of Everything Sent to and from the Server

By default, Aptimizer works to automatically reduce the size of content sent from the server.

- Text based objects are compressed using the gzip compression algorithm.
- JavaScript and style sheet files are minified by removing all redundant whitespace characters and comments.
- Redundant image information (such as metadata) is removed, and JPEG images are, by default, resampled to 85% of their original quality.

Cache as Much as Possible to Speed Up Repeat Views

Browsers can locally cache resources for faster reuse on subsequent Web page visits.

Aptimizer automatically applies cache headers to all resources, providing browsers with information needed to manage those resources. By default, Aptimizer applies different cache durations to resources depending on how they are referenced, as shown in the following table:

Condition	Cache for
Object is combined into a resource set	1 year
Object URL was versioned by Aptimizer	90 days
Original object URL was requested	1 hour

Public Proxy Caching

Optionally, Aptimizer can help leverage any intermediary caching proxy server, such as those commonly used in corporate environments and Internet service providers (ISPs). By using a specific attribute, Aptimizer will add "`cache-control: public`" headers to all or any specific resources.

Marking resources as proxy cacheable can provide a significant improvement in both first view and repeat view load times for users who access a Web page on a Web site other users have already accessed.

Reduce the Time It Takes for the Server to Respond to a Request

This rule applies to the time it takes for your server to start sending a HTML response to your client's browser. As previously mentioned, this can be up to 20% of the Web page load time.

Reducing this time is directly related to your server, application design and maintenance. Possible solutions can involve scaling up (achieved by adding system resources such as processor and memory to existing servers), scaling out (achieved by increasing the number of server nodes to distribute workload over more nodes), code profiling and tuning, database tuning, sharding (distributing the database over multiple servers), Web server tuning and more.

This rule applies to your Web server and application, thus outside the scope of this chapter.

Resource Naming and URL Versioning

Aptimizer will create a new identity for each **combined resource set** or **URL auto versioned resource** created in the process of accelerating your Web pages. This approach allows the Traffic Manager to inform the browser when a resource has changed rather than telling it to check back with the Traffic Manager every so often to see if it has changed. We do this by making the URL of the resource dependent on its content.

For example, a resource set could be named:

```
/aptimized-gif-kjmMLxSEdXYqXt3coAasow4F=-ds3432d.gif
```

and an auto-versioned URL could look like:

```
/images/aptimized-jpg-23dgdsds5563gfv=-P9vry3Uq22.jpg
```

The string of numbers and letters in these URLs represent a unique hash value of the settings, the content and any child resources that make up a particular resource. Whenever a setting is modified, any resources for which that setting applies will

automatically be recalculated. Thus, any change to the content of a resource will also cause this unique value to change.

Under normal circumstances, where the Traffic Manager is simply serving pages without auto-versioning enabled, the following typical data flow is in operation:

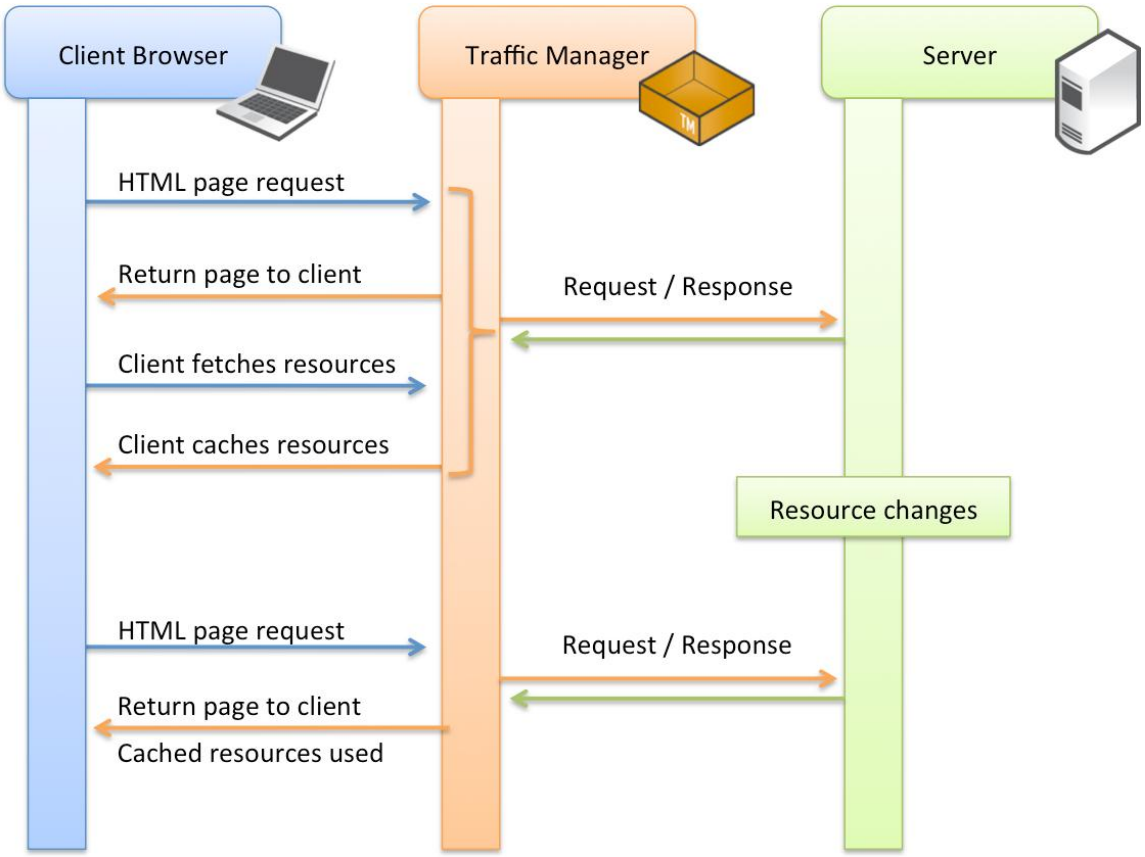


Fig 55. A typical data flow with auto-versioning disabled

1. The client requests an HTML page through a service running on the Traffic Manager.
2. The client receives the HTML page back, with links to regular resources (for example, style sheets, images, etc.).
3. The client fetches these resources.
4. The Traffic Manager serves them from the back-end server, possibly optimized depending on the Aptimizer settings used.
5. The client caches any available resources locally.
6. Some resource is changed on the back-end server for whatever reason (for example, bug fixing, a layout change, etc.).

7. The client requests another HTML page on the same website.
8. The client receives the HTML page from the Traffic Manager with a link to the same resources.
9. The client is unaware of the change on the server, and refers to its own locally cached (and unexpired) versions of any identified resources and does not re-request them from the Web service.

As shown in the last step, one essential flaw with this scenario is that the client will wait a period of time before re-requesting a cached resource, which may or may not have changed. The client could therefore be using an out of date copy of a resource until some event occurs to prompt the re-request.

With auto-versioning enabled, the Traffic Manager provides the assurance that the client will always be using the most up-to-date copy of a resource contained in a Web page. The following data flow demonstrates this:

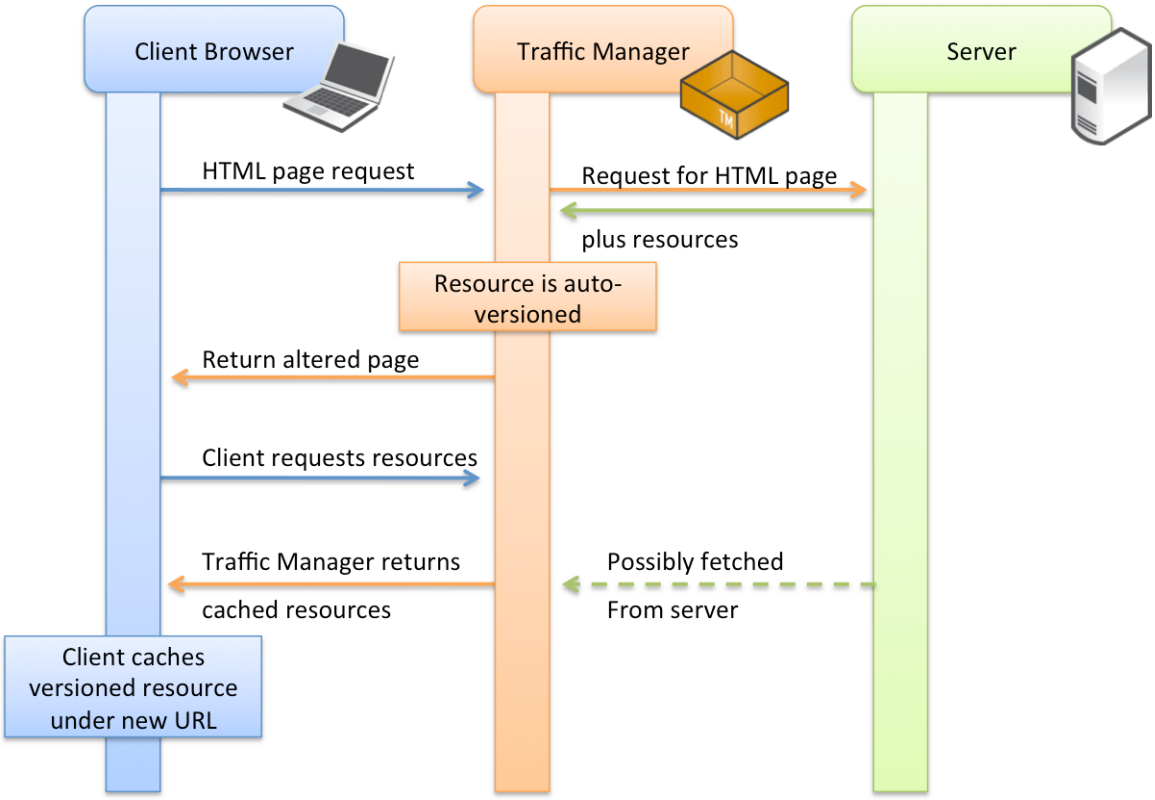


Fig 56. Typical data flow with auto-versioning enabled

1. The client requests an HTML page from a Web service provided through the Traffic Manager.
2. The Traffic Manager notices the HTML page references one or more resources.

3. The Traffic Manager fetches these resources (and possibly optimizes them, depending on the Aptimizer settings), examines the content and creates a hash value based upon it.
4. The Traffic Manager creates a new URL for such resources that includes the computed hash value, and replaces all references in the HTML page.
5. The altered page is returned to the client.
6. The client requests the in-page resources from the Traffic Manager, via the new URL.
7. The Traffic Manager serves the identified content from its cache (or fetches it from the back-end server if its not in the cache).
8. The client locally caches resources using the new URL.

Subsequent page requests are then served from the Traffic Manager with the test that the computed hash value of each resource has not changed (based on the resource content at the back end):

1. The client requests another HTML page.
2. The Traffic Manager observes that the contained resources have not changed on the back-end server and writes in the same computed hash-value URL as last time.
3. The client uses its cached copies of those resources.

Should a resource change on the back-end server, subsequent page requests will be affected by virtue of the computed hash-value being different. The following scenario occurs:

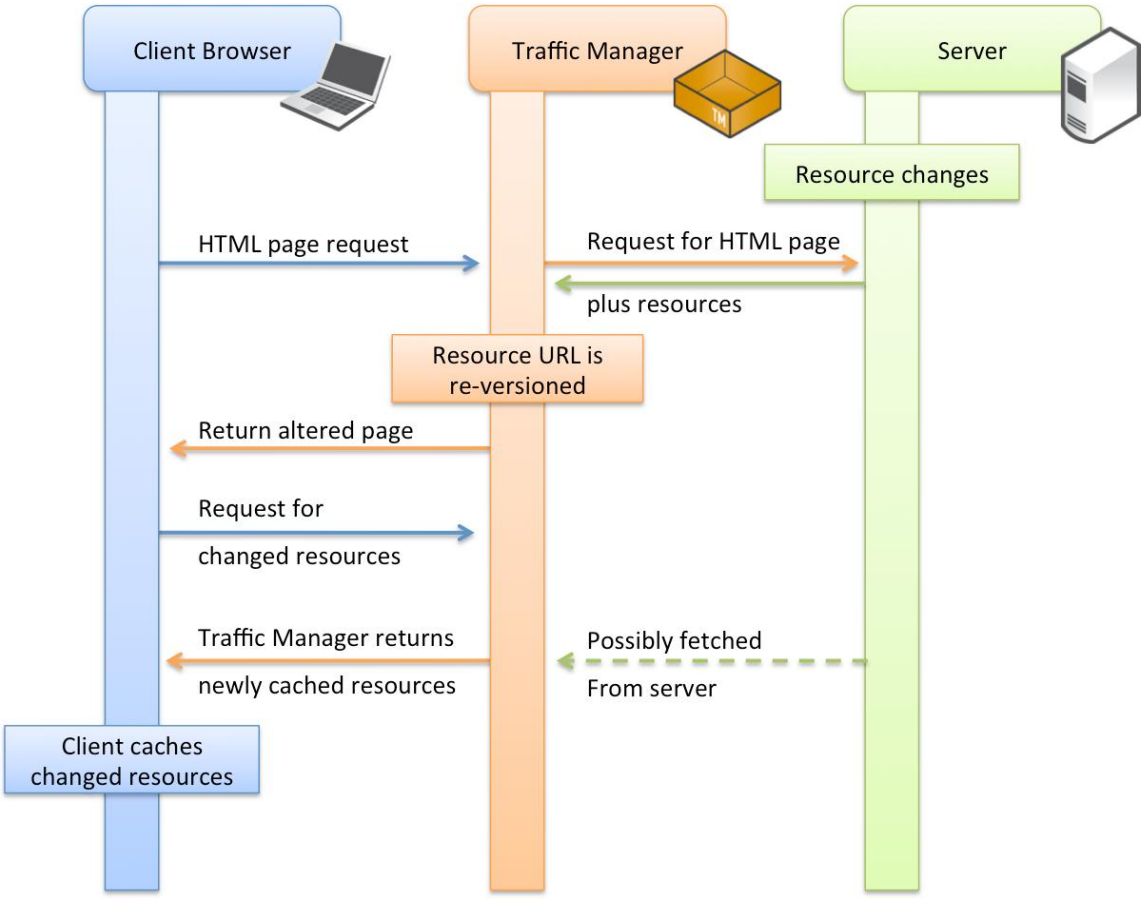


Fig 57. Typical data flow after a resource change

1. A resource change forces a re-compute of the content-based URLOne or more resources change on the back-end server.
2. The client requests another HTML page.
3. The Traffic Manager observes that the desired resource has changed on the back-end server and creates a new URL based on the new content - different to the previous one.
4. The client does not have a cache entry for the new URL, so it requests it from the Traffic Manager, which serves it back.
5. The client is immediately using the new version of any changed resource.

Using a Content Distribution Network

Content Distribution (or Delivery) Networks (CDNs) host some or all of your resources on servers that are geographically distributed, reducing latency between your Web server and the customer. For further information on CDNs, refer to http://en.wikipedia.org/wiki/Content_distribution_network.

You can leverage a CDN with Aptimizer using the **URL rewrite to CDN** configuration setting on the Aptimizer Profiles edit page (custom profiles only).

This setting instructs the Traffic Manager to rewrite the HTML for requests that match that rule so that resources are then downloaded from the domain provided by your CDN instead of your own Web server.

The Traffic Manager will automatically use the scheme (HTTP/HTTPS) of the incoming request when accessing a CDN host, but it is possible to manually specify the scheme and virtual path in the **URL rewrite to CDN** setting if required. This allows the scenario of switching all traffic to SSL if the target Web site is reverse proxied through another device that terminates the SSL connection.

Troubleshooting Aptimizer

Note: Certain UI controls and configuration options mentioned in this section do not apply if you are using Aptimizer Express.

Controlling Unexpected Behavior

When the Traffic Manager encounters problematic or non-standard user-defined content, the Aptimizer process could be forced into a failure state. If this occurs, the Traffic Manager records the failure with an error message in the event log and restarts the process.

In certain circumstances, error-prone content could force repeated failures and restarts of the Aptimizer process. This can hinder the overall performance of the Traffic Manager and make troubleshooting efforts more difficult. Therefore, the Traffic Manager provides a *watchdog* capability to stem the effect of frequent failures by disabling the Aptimizer process restart once a set limit has been reached.

Set your watchdog configuration in the **Aptimizer** section of the **System > Global Settings** page:

Setting	Description
aptimizer!watchdog_limit	This is the maximum number of times the Aptimizer process will be started or restarted within the interval defined by the `aptimizer!watchdog_interval` setting. If the process fails this many times on one Traffic Manager, Aptimizer will be disabled on all Traffic Managers in the cluster, and it must be restarted manually from the **System > Diagnose** page. If this limit is set to zero, Aptimizer will never be disabled.
aptimizer!watchdog_interval	The period of time (in seconds) after which a previous failure will no longer count towards the limit set by `aptimizer!watchdog_limit`.

Note: Most users should not have to use or modify these settings. If you do find that Aptimizer fails frequently, contact your support provider.

For details about other Aptimizer global settings, see "Other Configurable Global Settings" on page 323.

Interaction with Other Traffic Manager Functionality

In practice, enabling Aptimizer on your virtual server should present no interoperability issues with other Traffic Manager functionality. However, due to the nature of certain built-in protection and performance capabilities, it is important to understand the interactions between Aptimizer and these other features.

Problems with Dependent Requests

The following diagrams demonstrate the consequences of what are termed *dependent requests*. These are requests that follow a Web page request handled by the Traffic Manager, for dependent resources contained within that returned page.

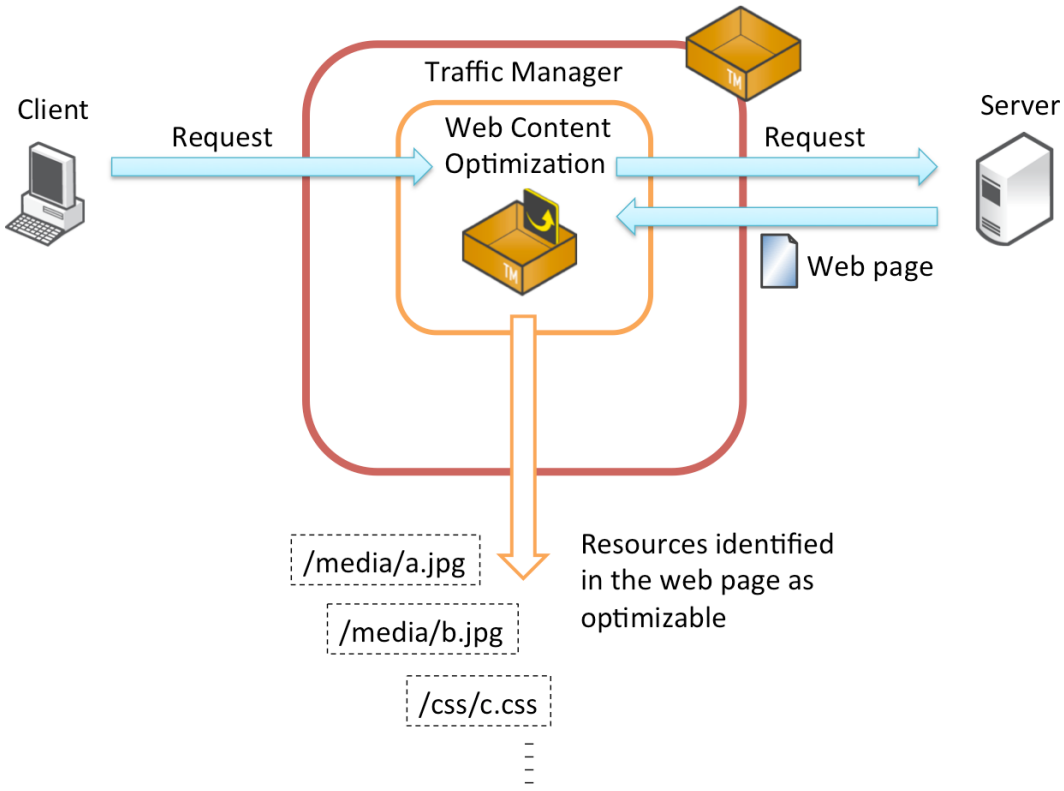


Fig 58. Identification of optimizable resources

Initially, the client requests a Web page for a service handled by the Traffic Manager. Aptimizer is enabled for this Web application and will attempt to identify resources that are potentially optimizable.

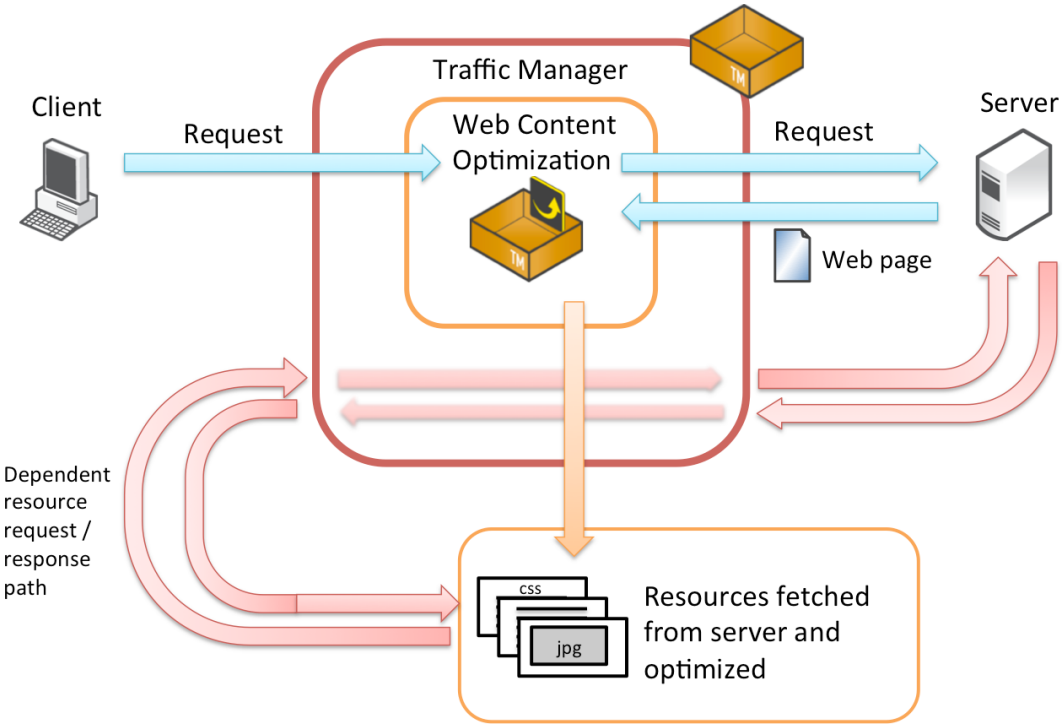


Fig 59. Dependent request/response model

The Traffic Manager may then need to make a number of additional *dependent requests* in order to locate and provide the various resources that have been stripped or optimized out in order to improve the page load time of your application.

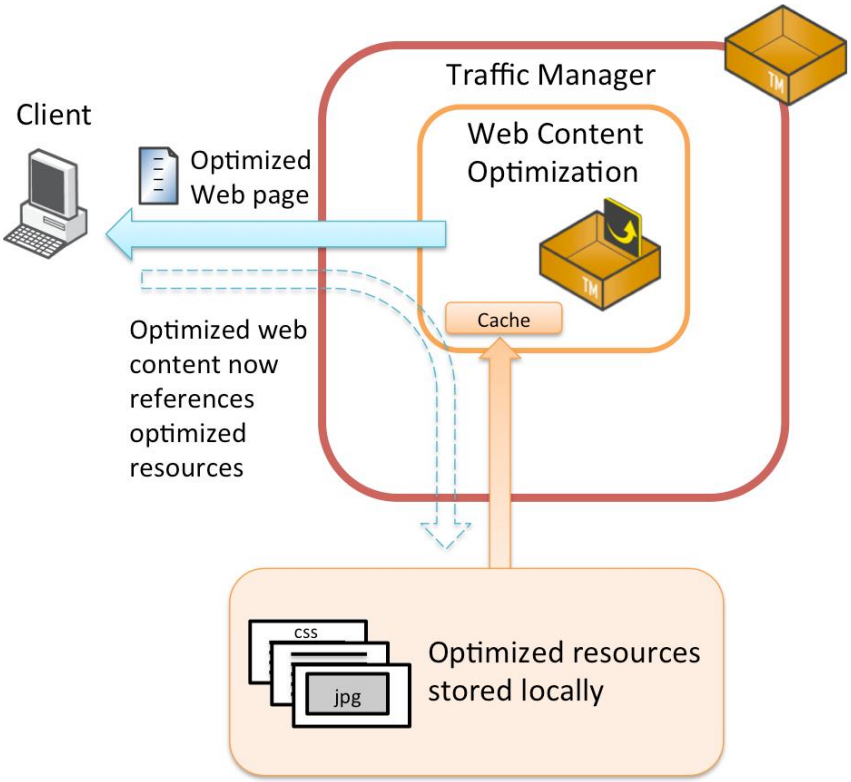


Fig 60. Optimized resources are cached and served by Aptimizer

After they have been fetched, suitable resources are optimized and cached locally by Aptimizer. Subsequent page requests contain modified URLs to optimized content, which can be served directly from the Traffic Manager.

Dependent requests are processed through the Traffic Manager as typical requests, with the usual processing applied. However these requests do not necessarily contain the same identifying headers, cookies and other elements of authentication as the requests and responses made directly through the Aptimizer, so can cause potentially unexpected results.

The following sections highlight the issues surrounding the fetching of these dependent requests.

Service Protection

The fetching of dependent requests could in some cases trigger Service Protection classes to be activated (since the requests will come in bursts from one of the Traffic Managers in your cluster). This can be prevented by adding the appropriate Traffic Manager IP addresses to the list of **Allowed IPs** (also known as *whitelisting*)

maintained within the **Access Restrictions** section of the class configuration. See CHAPTER 15 for further details about Service Protection features and capabilities.

TrafficScript Rules

The typical processing order for TrafficScript rules when Aptimizer is enabled runs as follows:

For a request:

Client request received -> TS request rules -> Aptimizer -> Sent to back-end node

For a response:

Response received from back end -> TS response rules -> Aptimizer -> Sent to client

---

Note: With respect to dependent requests, TrafficScript rules are run in the usual way.

TrafficScript rules that rely on the client IP address, browser headers, cookies, authentication data, or any other information pertaining to the client, could result in unexpected behavior due to the lack of such information in dependent requests.

Note also that use of the `http.stream.*` TrafficScript functions will prevent responses from being aptimized.

SSL Decryption

The Traffic Manager cannot be used to aptimize virtual servers that use SSL decryption and require client certificates. This is because a client certificate is not provided when Aptimizer initiates fetches for additional resources.

Authentication

In Express mode, the Traffic Manager cannot be used to aptimize any services employing authentication. In Enterprise mode, the Traffic Manager can aptimize services using NTLM. No other client authentication methods are supported.

Content Caching

Web content that is passed through the Aptimizer and can be considered to be in scope with Aptimizer will not be cached in the usual way within the regular Traffic Manager Web cache. Instead, it will be held and served from a dedicated Aptimizer cache, designed to maintain local copies of optimized Web content/resources. This behavior is controlled via the **Caching** tab on the Aptimizer Profile edit page.

Runtime Errors

Runtime errors are generally server-related. You can get these errors due to server misconfiguration or third party interference (routers, firewalls, load balancers). In

general you might not be able to access the Web site at all, of if you can access the Web site you might experience missing resources.

Symptom: An error page is displayed only when Aptimizer is enabled; resources are served with a 400 or 404 http status; the server takes a long time to return Web pages even though the application pool is warm.

> **Problem**: A URL rewriter or other HttpModule is attempting to handle the Aptimizer resource set URLs.
>
> **Resolution**: Exclude URLs containing `/aptimized-gif-*.gif` from URL rewriting or processing by another HttpModule.

Symptom: The Web service works when accessed locally but not when accessed from external clients.

> **Problem**: A proxy or upstream device is blocking or altering `/aptimized-gif-*.gif` URLs causing them to become invalid.
>
> **Resolution**: Allow requests for `/aptimized-*` to pass through to the host server unmodified.

Symptom: Cannot see the comment "`<!-- Optimized for speed - Stingray Aptimizer -->`" in the HTML.

> **Problem**: When you use your browser's "View HTML Source" the comment "`<!-- Optimized for speed - Stingray Aptimizer -->`" is not present, indicating the Web page is not being accelerated.
>
> **Resolutions**:
>
> - Make sure Aptimizer is enabled for the specific virtual server.
> - Make sure the URL is not being excluded by a rule set.
> - If Aptimizer is running in Stealth Mode use the `?aptimizer=on` query string in the URL request.
> - If the URL is not being excluded, Aptimizer is fully enabled, make sure `?aptimizer=off` is not used in the URL request.

Image Errors

You will have image errors if your Web pages do not display images in their specified position or display images with incorrect size. Image errors can also be "invisible", as it is the case when Aptimizer creates new resource sets for every new visit to the page.

You can spot the problem with resource set creation by inspecting the elements on the page. You should see the most common elements being created once and served from the cache multiple times.

Symptom: A new sprite set seems to be created every time I visit the page.

> **Problem:** The list of images in the page is different every time, which causes Aptimizer to create a new sprite set every page visit. This can cause excessive processor utilization, high cache memory usage, and overall reduced performance on the server.

Resolution: Add a URL rule to include the dynamic images in an uncombined form (disable the setting **Combine HTML images into image sprites** in the **Images** tab).

Symptom: Images on the page are losing their padding or out of alignment on the page.

> **Problem:** Padding is being used to space the image from its neighboring UI elements

Resolution: Add a URL rule to include the dynamic images in an uncombined form (for example, disable the setting **Combine HTML images into image sprites** in the **Images** tab).

Symptom: One of the images appears much larger when Aptimizer is enabled.

> **Problem:** The container that holds the image uses style sheet to resize the image from its native dimensions.
>
> **Resolutions**:
>
> - Modify the page template to include a width and height attribute on the <img> element. This lets Aptimizer know what size the image should be displayed at, and also helps the browser calculate the page layout more quickly, resulting in a faster rendering time.
> - Add a URL rule to include the dynamic images in an uncombined form (for example, disable the setting **Combine HTML images into image sprites** in the **Images** tab).

Symptom: Acceleration is working but there is a layout problem.

> **Problem**: This is normally style sheet or HTML image related.

Resolution: Reduce the number of style sheet objects and HTML images. This problem typically requires the use of acceleration rules to work around whatever technique is causing this symptom to appear.

CSS Errors

You could have image errors if your Web pages do not display images in their specified position or display images with incorrect size. Image errors can also be "invisible", as it is the case when Aptimizer creates new resource sets for every new visit to the page.

You can spot the problem with resource set creation by inspecting the elements on the page. You should see the most common elements being created once and served from the cache multiple times.

Symptom: Internet Explorer does not apply background images defined in a style sheet.

Problem: The cause is the security policy that Internet Explorer is enforcing. If an HTML page on one domain references a resource on a different domain and that resource type is rfc822 then the browser will load it but not process it since it does not know if it can be trusted. External resources on the same TLD as the page are implicitly trusted.

Resolution: Change the CDN hostname to be a subdomain of the main Web site so that resources are implicitly trusted.

JavaScript Errors

It is important to ensure any JavaScript errors, as detected by most modern browsers, do not exist before enabling Aptimizer. This will help determine if these problems are caused by any of the introduced optimizations. If in doubt, request the page again with Aptimizer disabled. This will bypass the accelerator and you can then check if JavaScript errors exist regardless of the optimizations.

On Internet Explorer 9 you have access to Developer Tools and debug information. On Firefox and Chrome you can install the Firebug add-on and access similar information. Most browsers offer an identifying message/notification when JavaScript errors are encountered regardless of installed tools and plug-ins. This message is usually helpful in disclosing which script is causing problems and frequently also identifies the specific position or line number of the erroneous section of code.

Debugging JavaScript Errors

Because Aptimizer allows you to apply rules to specific URLs, you can switch off certain optimizations for individual (or a group of) resources without compromising the default settings for your Web site.

To resolve JavaScript issues with Aptimizer enabled you can perform one or more of the following:

- Disable the **JavaScript optimization** option on the **JavaScript** tab of any matching rules
- Identify which script is breaking and create a URL rule to exclude it completely from optimization
- You can verify Aptimizer is causing JavaScript errors quickly by excluding **all** JavaScript optimization on your web site. Create a temporary URL rule to exclude all optimization for any URLs containing .js.

Common JavaScript Errors

Symptoms: Scripts execute with unpredictable results.

> **Problem**: Errors due to an external script reference having been moved in the page.
>
> **Resolution**: If order-dependent code exists between the external script and other script files or an inline script block, then this may prevent normal functioning of the site. Best practice for building JavaScript libraries is to treat them as libraries of functions, and not execute any immediate code. This allows script to be combined to the maximum extent possible. To work around poorly designed script you can instruct Aptimizer to exclude them entirely via a suitable URL rule.

Symptoms: Scripts fail to load and execute.

> **Problem**: JavaScript frameworks that dynamically load script libraries into the page.
>
> **Resolution**: These frameworks typically look for a reference to one of the scripts that they use, or will use some sort of registration function to inject the scripts that should be loaded. Since Aptimizer refers to combined scripts using a surrogate URL, this can confuse frameworks that behave like this. The solution in this case is to either exclude these scripts, or transform the loading code such that it understands Aptimizer modifications.

Symptom: Some JavaScript is not executed on a Web page when using the JavaScript async option.

Problem: Some ad blocking software will incorrectly filter out the Aptimizer code injected to asynchronously load and execute JavaScript.

Resolution: Use a URL rule to exclude required scripts from the async code, or disable the option to load and execute JavaScript in asynchronous mode.

Other Configurable Global Settings

The Traffic Manager additionally provides the following configurable global settings in the **Aptimizer** section of the **System > Global Settings** page:

Setting	Description
aptimizer!max_original_content_buffer_size	The maximum content size of a single back-end response that can undergo Aptimizer optimization. Responses larger than this are returned to the client un-optimized. **Note**: If the back-end response is compressed, this setting pertains to the compressed size of the response body.
aptimizer!max_dependent_fetch_size	The maximum the size of a dependent resource (a resource that Aptimizer has requested from the back end, as a piece of content currently undergoing optimization depends on it) that can be sent to, and processed by, the Aptimizer sub-process. Any resource larger than this value is always ignored by Aptimizer and returned to the client un-optimized. Set this field to zero to disable the limit.

You do not normally need to alter these settings. The Traffic Manager uses default values that typically provide a good level of performance on all systems.

CHAPTER 21 Event Handling and Alerts

The event handling capability allows the administrator to configure precisely what actions the Traffic Manager should take if particular events occur. Event handling behavior can be configured in the **System** section of the Administration Interface, under the **Alerting** tab.

Overview

An Event Handler specifies the actions that should be performed when an event of a particular type occurs. Event Handlers are configured on the **Alerting** page:

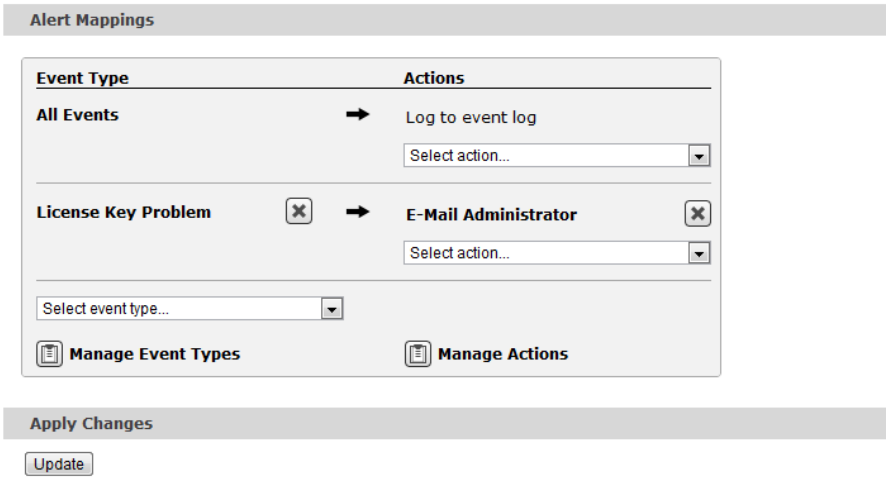


Fig 61. Configuring event handlers

The Traffic Manager contains one built-in event handler that causes all events to be written to the global Event Log. This built-in handler and its "Log all Events" action cannot be deleted, but you can add additional actions if required.

You can add additional event handlers, such as the handler illustrated above that sends an email when there is a problem with a license key. Start by selecting the Event Type from the drop-down box, then select the actions that should be performed when an event of that type occurs.

Note: when you create an additional Event Handler, you can add the internal action 'Bypass event log'. Any events that are processed by that Event Handler will not be logged to the global Event Log:

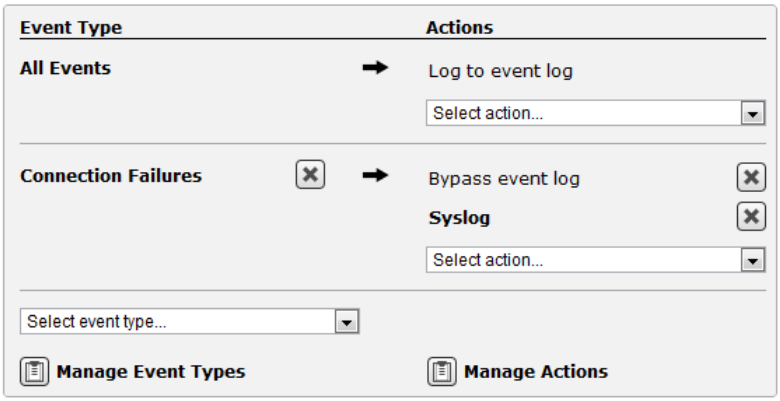


Fig 62. Events that match the Connection Failures Event Type will be handled by the Syslog action, and will not be logged to the Traffic Manager Event Log.

Event Types

The Traffic Manager can respond to a wide variety of events, such as node failures, Traffic IP address transfers, Service Level failures and custom events raised from a rule. For convenience, individual events are grouped into a number of pre-defined types, and you can create new Event Types as required.

Event Type	Description
All Events	Includes all of the events that the Traffic Manager can detect; a built-in handler will log all events that match this type to the global Event Log.
Critical Problem Occurred	Includes all of the events that indicate a serious problem, including internal failures, node failures, IP transfers (because a Traffic Manager has failed) and other hardware problems.
Critical Problem is Resolved	Includes events that indicate that a problem that was raised in the 'Critical Problem Occurred' Event Type has been resolved; for example, a failed node has started working again. If you configure the Traffic Manager to send an email message when a problem occurs, it is sensible to also configure the Traffic Manager to send an email message if the problem is resolved and intervention is no longer necessary.

Event Type	Description
Default Events	Included for compatibility with earlier versions of the Traffic Manager; it includes all of the events that would have caused the Traffic Manager to send an email message or raise an SNMP trap. When the Traffic Manager is upgraded from an earlier version, the upgrade process will add an Event Handler to preserve the earlier behavior if necessary.
Infrastructure Problem	These events are raised if the Traffic Manager detects a problem with an item of infrastructure; a server becomes un-contactable for example.
Service Problem	These events are raised if a problem is detected that could impact the successful delivery of a service.

You can view the Event Types using the **Manage Event Types** button on the **Alerting** page.

Creating New Event Types

You cannot modify the built-in Event Types, but you can make a copy that can be edited, or you can create new Event Types from scratch.

For example, suppose that you use the Traffic Manager to manage several services, and you wish to create an Event Type that is matched when a particular service fails. You could then use that Event Type to create an event handler that emails the individual responsible for that particular service.

To do this, create a copy of the 'Service Failed' Event Type, and edit the copy so that it is only matched when particular virtual servers, pools and nodes raise events:

1. Click Manage Event Types, and click the 'Service Failed' Event Type.
2. You cannot modify this Event Type, but you can make a copy; copy it to an Event Type called 'Custom Service Failed'.
3. You will be presented with the Edit page for your new Event Type. Events are arranged in a tree structure; you can expand the nodes in the tree to see all of the events that the Traffic Manager can detect. The particular events that the Event Type matches are opened so that they can be easily seen.

4. Edit the new Event Type. You may want to change the list of health monitors, nodes, pools, rules, virtual servers and other objects that raise events so that the Event Type is only matched when the objects relating to your specific service raise events.
5. Apply your changes, then create an appropriate Event Handler for the 'Custom Service Failed' Event Type.

Actions

When an event handler is triggered, the Traffic Manager will invoke the actions configured for that handler.

Note: The Traffic Manager contains three built-in actions: E-Mail, SNMP Trap and Syslog. Note that on a new install of the Traffic Manager, the E-Mail and SNMP Trap actions are not completely configured (you will need to edit them to specify the required settings). However, when the Traffic Manager is upgraded from an earlier version, these settings are preserved.

You may create additional actions of the following types:

Type	Description
E-Mail	This action sends a report of the event(s) that triggered the handler to the email addresses configured in the action. You can create several E-Mail actions with different email addresses, and use this to configure the Traffic Manager to email different individuals for different Event Types.
Log to external file	Events are always logged to the global event log, but this action can be used to additionally log events to a named log file. The format used is the same as the format used in the global event log. This action is useful when creating a dedicated log of particular events. **Note:** Traffic Manager appliance/cloud users: When using this facility on a virtual or cloud instance, you should ensure your file is kept within ZEUSHOME. When applying an upgrade, all user-created files and directories outside of ZEUSHOME are likely to be lost.

Type	Description
Log to syslog	This action will send a message to the syslog daemon on the named server. This action can be used to send particular events to an external monitoring and management tool that can accept syslog messages.
Program	This action runs a script or other program that you have uploaded to the Traffic Manager system using the **Upload and Manage Programs** button on the action's **Edit** page. Refer to the *Calling a Program or Script* section of CHAPTER 21 for an example program.
SNMP Notify/Trap	This action sends an SNMP Notify or Trap to the named SNMP receiver application on the host specified. Traps are defined in the Traffic Manager MIB, which you can download from the action's **Edit** page. The Traffic Manager supports SNMP v1, v2c and v3. The configurable settings for this action are dependent on the version you specify on the Edit Action page, and directly mirror the available settings on the **System > SNMP** page (SNMP command responder options). Refer to the *Monitoring Performance Using SNMP* section of CHAPTER 7 for more details.
SOAP Callback	This action sends a SOAP message to the named server. Refer to the *Sending a SOAP Message* section of CHAPTER 21 for an example of a SOAP action and a server that handles it.

On the **Alerting** page, click the **Manage Actions** link to edit actions and create new actions.

You can also apply the internal Action 'Bypass event log' to prevent events being logged to the global event log.

Testing Actions

When you create or edit an action, you can invoke it immediately with a 'test' event. This is a useful way of testing the correct configuration and operation of your actions.

Configuring an Event Handler

Before you configure an event handler, you will need to create an Event Type that matches the events you would like to trigger the event handler, and create an Action that performs the correct task, such as sending an email.

The event handler links the Event Type with one or more actions that should be invoked when an event that matches the Event Type is raised.

To configure an event handler, go to the **System > Alerting** page. Select the Event Type from the drop-down list to create a new event handler, then select the Actions you wish to be invoked from the drop-down in the entry for that event handler.

Press 'Update' when you have completed your changes.

Duplicate Events

An event may match several Event Types, and therefore trigger several event handlers. If the same action is configured on each of these handlers, it will be invoked multiple times.

To reduce the volume of small emails sent, the E-Mail action is not triggered immediately, but will gather email messages and send them at 30 second intervals. You can configure this behavior of the email action in the **System > Global Settings** page, in the **Other Settings** section.

If a minor event is raised several times in quick succession, the Traffic Manager will suppress the repeated events. Major events are never suppressed in this way.

Custom Actions

Calling a Program or Script

The Program action is used to run a program or script that you have uploaded to the Traffic Manager. Programs and scripts are uploaded using the **Catalogs > Extra Files > Action Programs** page.

When this action is invoked, the Traffic Manager will execute the program in the background, passing it command line arguments that identify the event that triggered the action.

The Traffic Manager passes the following arguments to the program or script:

- Any command line arguments you have configured in the action (optional).

- An argument named `eventtype` that provides the name of the event handler that invoked the action that executed the program.
- The event description; this is in exactly the same format as the log line for the event that is written to the event log.

For example, if a TrafficScript rule raises an event named TestEvent, and this is caught by an event handler named 'My Event Handler' which invokes a Program action, the program will be called as follows:

```
programname "--eventtype=My Event Handler" \
   "INFO trafficscript/TestEvent CustomEvent
rules/<rulename> \
   vservers/<vservername> <description>"
```

Note that the script is given two arguments (linebreaks have been inserted for readability).

Sending a SOAP Message

A SOAP action sends a SOAP request to the proxy configured in the action. The SOAP message conforms to the WSDL specification in:

`ZEUSHOME/zxtm/etc/wsdl/AlertCallback.wsdl`

You can download this file from the **Actions > Edit** page when you edit a SOAP action. This specification is documented in the Traffic Manager Control API documentation.

The SOAP request uses the interface `http://soap.zeus.com/zxtm/1.0/AlertCallback`, and invokes the method `eventOccurred`, passing it the following arguments:

- The name of the Traffic Manager machine that raised the event (type `xsd:string`)
- The time of the event, in the `xsd:dateTime` format[10]
- The severity of the event (of the enumeration type `AlertCallback.Severity`)
- The tag that identifies the event (of the enumeration type `AlertCallback.Tag`)
- An array of tags; this is empty and reserved for future use

[10] http://www.w3.org/TR/xmlschema-2/#dateTime

- An array of objects (type `AlertCallback.ObjectArray`); these objects (type `AlertCallback.Object`) describe the object that was responsible for raising the event (such as a node or rule)

  Some events will reference multiple objects. For example, an event raised from a TrafficScript rule will contain three objects that identify the event name, the rule and the virtual server running the rule;

- A human-readable description of the event (type `xsd:string`). This is predefined for most events, but when an event is raised by the TrafficScript function `event.emit()`, the description contains the message provided by TrafficScript

- Additional data provided by the action (type `xsd:string`)

- The name of the Event Handler that invoked the action

The following Perl CGI script implements a simple SOAP consumer for the AlertCallback messages; this responder logs the details of the message to a local file. You can install this CGI script on a suitable Web server (equipped with Perl and the SOAP::Lite Perl modules), then configure a SOAP action, setting the proxy value to the URL of the CGI script:

```
#!/usr/bin/perl

package AlertCallback;

sub _addSerializerForEnum($;$)
{
   my ( $xsdtype, $ns_prefix ) = @_;
   $ns_prefix = "zeusns" unless $ns_prefix;

   $func = "SOAP::Deserializer::as_${xsdtype}";
   *$func = sub { return $_[1] };
}

BEGIN {
   _addSerializerForEnum( "AlertCallback.Tag" );
   _addSerializerForEnum( "AlertCallback.ObjectType" );
   _addSerializerForEnum( "AlertCallback.Severity" );
}


sub eventOccurred($$$$$$$$)
{
   my $self = shift;
```

```
   my( $stm, $time, $severity, $tag, $tags, $objects,
      $desc, $additional, $handler ) = @_;

   open LOG, ">>events.log" or die( "Couldn't open log
events.log: $!" );

   print LOG "$stm - $time - $severity: $tag [";
   print LOG join ', ', map { "$_->{type}:$_->{name}" }
@$objects;
   print LOG "] $desc ($additional); Event handler:
$handler\n";

   close LOG;
}

package main;

use SOAP::Transport::HTTP;

SOAP::Transport::HTTP::CGI
 -> dispatch_with(
    { "http://soap.zeus.com/zxtm/1.0/AlertCallback/" =>
'AlertCallback' } )
 -> handle;
```

If your action is invoked by multiple different events, you may wish to determine the precise event based on the tag value in the SOAP message. You can do so by matching the description of the event in the event tree (when you configure a custom Event Type) with the description in the WSDL file to determine the value of the `AlertCallback.Tag`.

Raising Events from TrafficScript or Java Extensions

You may raise an event from within a TrafficScript rule or a Java Extension. The appropriate event handlers will be run asynchronously, in the background, and the TrafficScript rule or Java Extension will not block.

Events are raised using the `event.emit( event_name, event_message )` function (TrafficScript) or the `emitEvent( event_name, event_message )` function (Java).

To invoke an action based on an event raised using `event.emit()` or `emitEvent()`, you need to create an Event Type that includes Custom Events. You can choose to match all custom events, or just specific events:

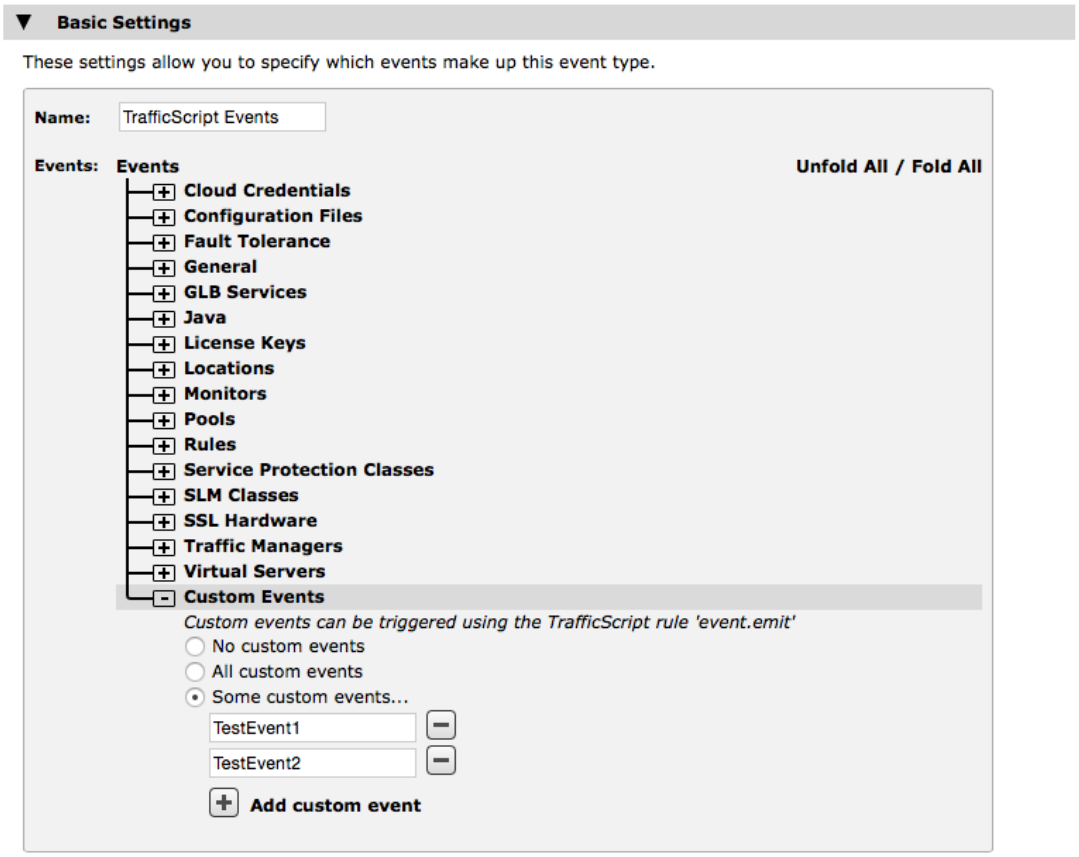


Fig 63. An Event Type that matches the TrafficScript events 'TestEvent1' and 'TestEvent2'

The names of the specific events in the Event Type should match the name used when the event was raised in the TrafficScript rule:

```
# raise an event named "TestEvent1", to be caught by an
Event
# Type containing a custom event called "TestEvent1".

event.emit( "TestEvent1", "The request was ".$url );
```

Example

A simple intrusion detection system may wish to ban all remote clients that attempt to access a privileged URL:

```
# A TrafficScript rule that detects unauthorized
# accesses and raises an event.

$path = http.getPath();

if( string.StartsWith( $path, "/secure/admin" ) ) {
```

```
   $ip = request.getRemoteIP();
   if( !string.ipMaskMatch( $ip, "10.0.0.0/8" ) ) {
      event.emit( "UnauthorizedIP", "IP: ".$ip );
   }
}
```

You should create an Event Type that contains the custom event named "UnauthorizedIP", and an event handler based on that Event Type.

The event handler could then invoke a custom program action that updates the intrusion detection rules to ban connections from that IP:

```
#!/usr/bin/perl -w

# An Action script to process event messages raised by
# the above TrafficScript rule.

my $logLine = $ARGV[1] or exit;

$logLine =~ /IP:([\d]+\.[\d]+\.[\d]+\.[\d]+)/;
my $ip = $1;

open OUT, ">>/tmp/events.log" or die $!;
print OUT "The bad IP was $ip\n";
close OUT;

# reconfigure upstream firewall, or the
# Traffic Manager's banned IP list to ban access
# from $ip
```

CHAPTER 22 Configuring System Level Settings

Note: This chapter applies only to certain Traffic Manager product variants. Refer to your product specifications for further details.

Traffic Manager virtual appliances, cloud instances, and certain software variants contain a number of additional configuration pages to manage settings such as network configuration, and time and date. On these variants, the Traffic Manager Administration interface performs a number of additional system checks and tests normally handled at the operating system level, such as checking the integrity of various operating system components and features.

For initial setup and configuration instructions, including details about how to reset the Traffic Manager to its initial factory configuration, see the "Installation and Getting Started Guide" applicable to your product variant.

Network Configuration

Note: The features described in this section might vary depending on your product variant. Refer to your product specifications for further details.

Your Traffic Manager instance contains a number of network interfaces, and requires a hostname to identify it. The **Networking** page in the **System** section of the Admin Interface allows you to configure these settings, along with DNS, Routing and IP forwarding and NAT.

Note: The settings you configure on this page apply only to the local Traffic Manager instance whose Admin UI you are accessing. The exception to this is **Return Path Routing**, which is applied across your Traffic Manager cluster. If you want to inspect or configure the network settings for a different Traffic Manager in your cluster, use the Admin UI for that instance.

Configuring the Hostname and IP Addresses

The hostname uniquely identifies your Traffic Manager system within your cluster, and the IP addresses identify each active network interface on the system. If you do not use a management port, the hostname you select must resolve to a permanent IP address on the Traffic Manager system, so you will need to configure your local DNS accordingly.

Typically, the first network interface (eth0) is designated as the management port and the hostname you specify does not need to resolve to the IP address on that machine. The Traffic Manager will only accept and transmit management

information on the management IP, and it should be connected to a secure network, separate from networks that route public internal or external traffic. Additional network interfaces are used to manage external or internal traffic.

If you chose not to use a management port, then the Traffic Manager will accept management traffic on all interfaces, and the hostname you specify must resolve to one of the IP addresses active on the Traffic Manager system.

The "Installation and Getting Started Guide" applicable to your product variant describes which interfaces are present on your Traffic Manager, and how they are numbered.

Configuring Networking

By default, the Traffic Manager auto-negotiates network settings with the switch each interface is connected to.

It is sometimes necessary to manually specify the interface settings. For example, many switches may fail to negotiate the fastest speed they support. Note that 1Gb speeds are auto-negotiated and cannot be forced.

Trunking/Bonding

If you configure the same IP address on multiple physical network interfaces, the Traffic Manager makes the interfaces members of a trunk (applicable only to virtual appliances). For this to work, your switch must support IEEE 802.3ad, which might require you to reconfigure it. Your switch might refer to this feature as link aggregation, bonding or trunking.

Configuring VLANs

Note: It is not possible to configure VLANs through the Traffic Manager Admin UI for virtual appliances running on the Microsoft Hyper-V hypervisor. If you need to enable VLANs in this scenario, you should configure them directly through the Hyper-V Manager application on your Windows Server workstation. Refer to your Hyper-V documentation or support provider for further information.

You can use the Traffic Manager Admin UI to configure VLANs on any of the network interfaces or bonded/trunked interfaces on your Traffic Manager. This configuration is performed from the **System > Networking** page:

1. Add a Virtual LAN interface to the appropriate physical interface:

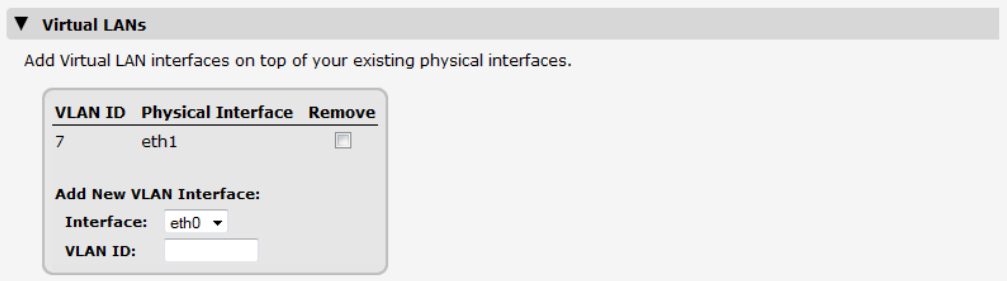


Fig 64. Virtual LAN configuration

2. Configure the networking (IP address and subnet) on the new VLAN:

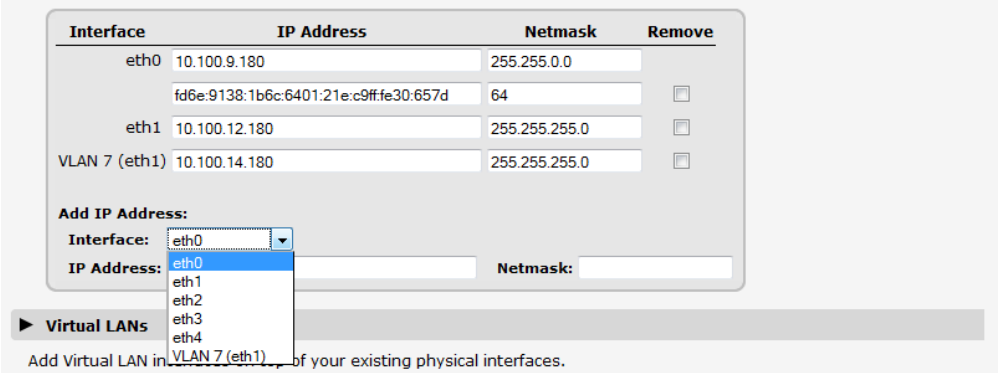


Fig 65. Virtual LAN Network settings

Any traffic that is routed out through the VLAN interface is tagged with the VLAN ID.

Configuring Your DNS Settings

The Traffic Manager will query a local DNS server for inter-cluster communications, when resolving node names to IP addresses, when communicating with external services and whenever access restrictions or other configuration required it to resolve the IP address of incoming network connections.

You will need to configure your Traffic Manager system with the IP addresses of one or more local DNS servers. These are normally configured when you first install the Traffic Manager, but you can edit this configuration and add fixed host-to-ip-address mappings using the **DNS** part of the **Networking** page in the Admin Server.

Configuring Routing

Your Traffic Manager will route traffic through the most appropriate network interface according to the IP addresses you have assigned to these interfaces. You should also configure a default gateway to which the Traffic Manager can forward non-local network traffic.

Important: Your default gateway is used in network connectivity tests, as described in the *Configuring Fault-Tolerance* section of CHAPTER 6. It should reside on the network used for external (incoming) traffic. Do not specify a gateway that resides on the management network, or configure the **flipper!frontend_check_addrs** with more appropriate IP addresses.

If necessary, you can manually add additional routes to non-local networks when the default gateway is not appropriate. You must specify the non-local network (destination and netmask), the local gateway to that network (optional) and the interface to use to access the non-local network.

These settings are managed in the **Routing** part of the **Networking** configuration page in the Admin Server.

Configuring Return Path Routing

Note: The settings in this section are not unique to a single Traffic Manager and are replicated to all other Traffic Manager instances in your cluster.

Typically, you do not care which route data packets take to get to their destination after leaving your network, as long as they arrive.

Traditional routing algorithms examine the destination address of a data packet and send it to the router that is believed to be the lowest-cost path to that destination. In cases in which two or more routes exist between your Traffic Manager and the Internet, the route from client to server can be different from the route from server to client. Normally this does not matter, provided that both are still receiving the packets they expected.

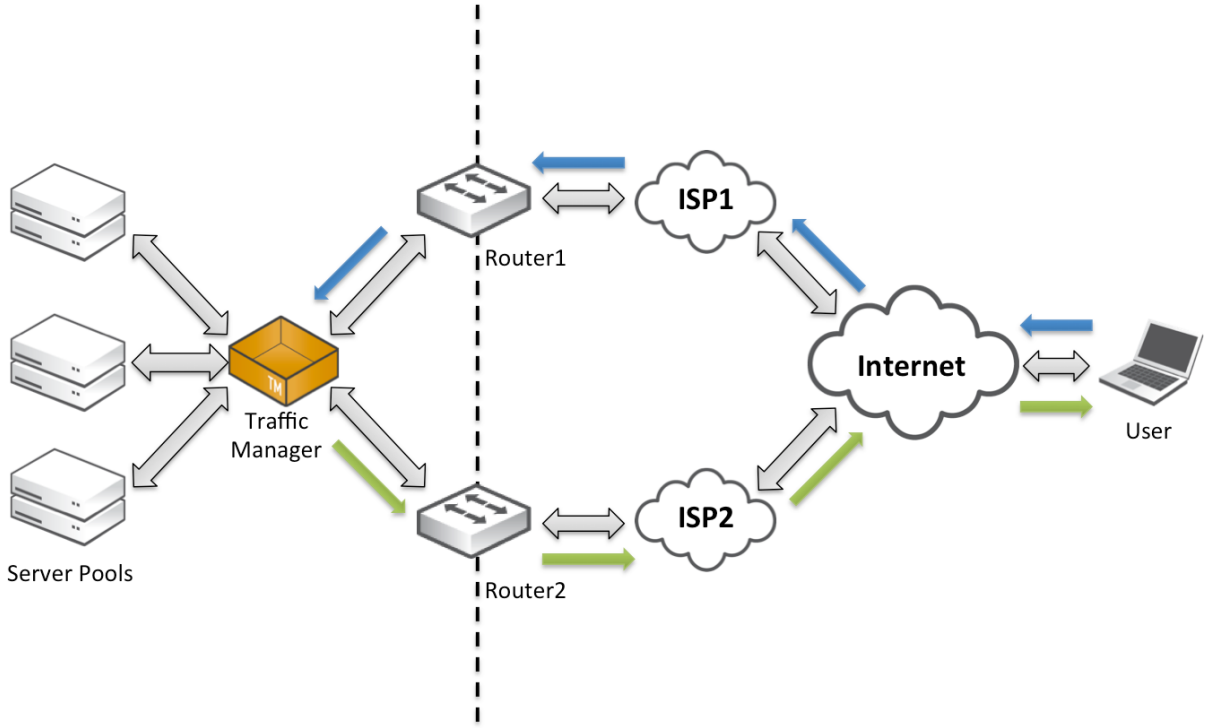


Fig 66. A typical lowest-cost route

However, in some cases you might need the outbound path to correspond to the inbound path. For example, if there is a *stateful* firewall or NAT router in the path, the device needs to handle packets in both directions; otherwise, it drops packets that do not belong to a connection it knows about. Alternatively, you could be using two ISPs for load balancing (in addition to redundancy). If all server-to-client traffic flows through the same ISP, it can cause an unbalanced load.

One solution to this problem is to install a gateway router as the default gateway for the Traffic Manager. The gateway router forwards packets to the correct ISP using its own connection tracking.

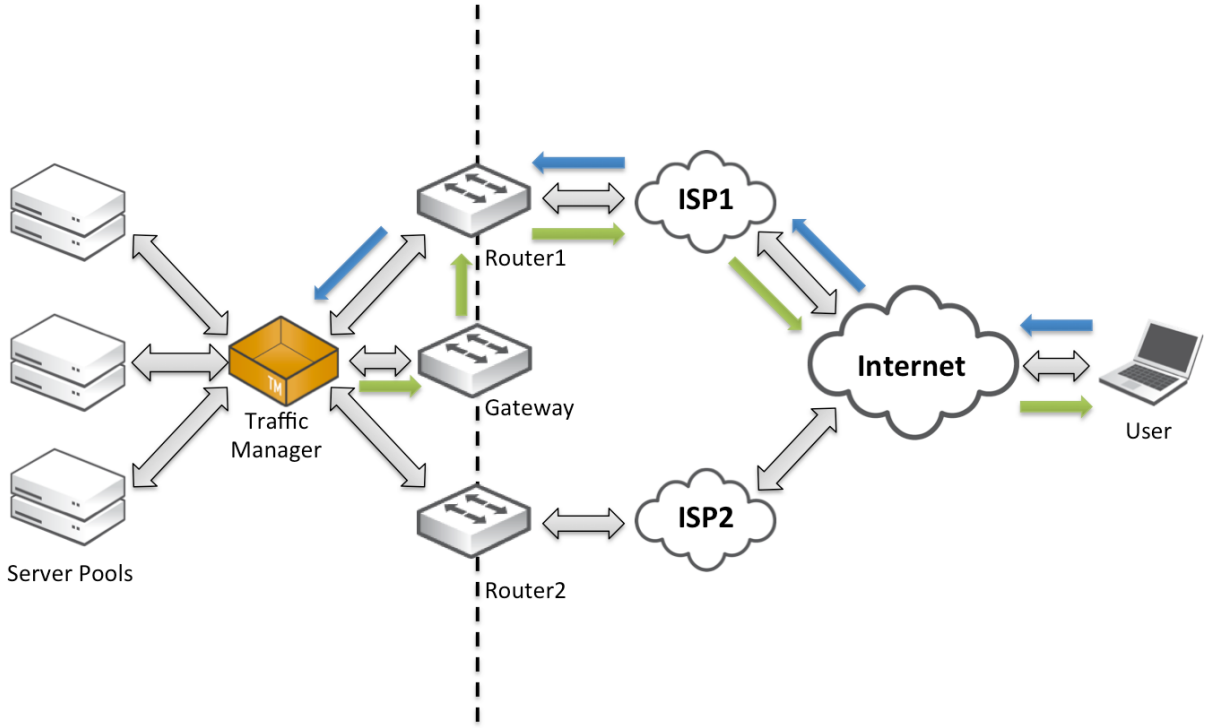


Fig 67. Employing a gateway router to perform connection tracking

If this solution is unachievable in your network infrastructure, you can use the Return Path Routing feature of the Traffic Manager appliance to assist in providing the correct return route.

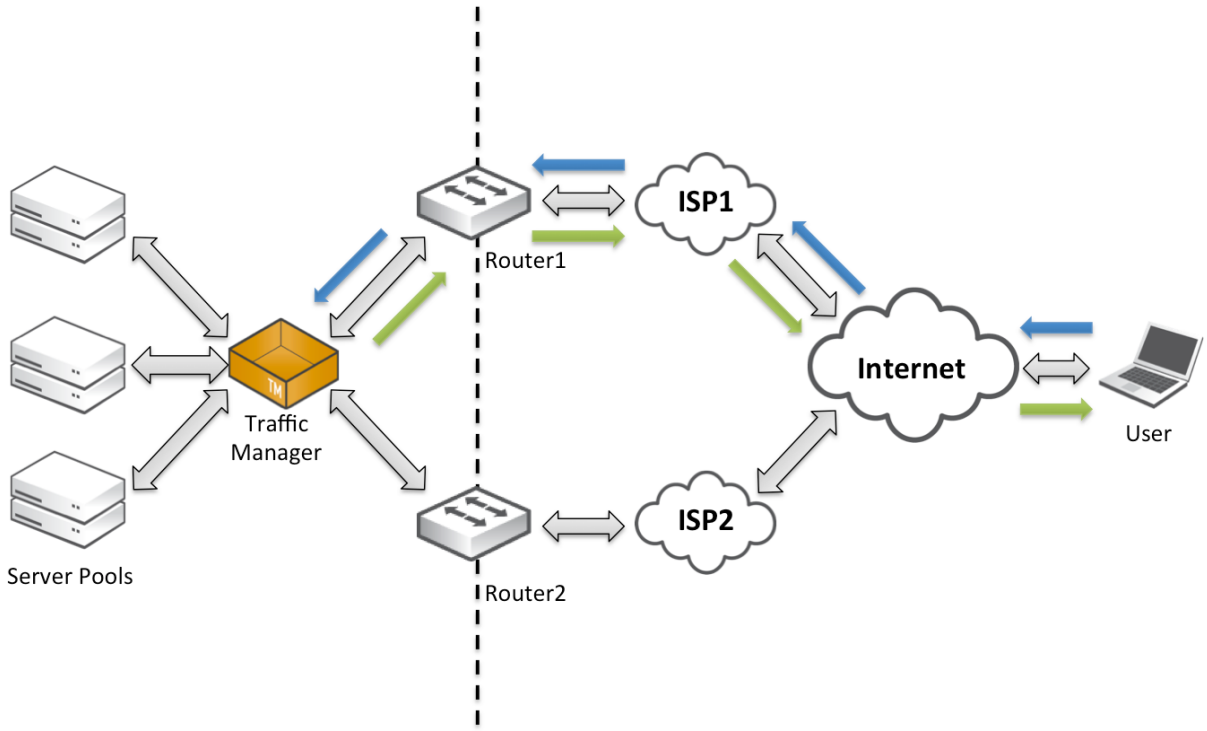


Fig 68. Return path routing is enabled

Note: The Traffic Manager can provide only return path routing back through an initial router hop. You are responsible for ensuring that your network is capable of providing the necessary onward routing back to the client.

The Traffic Manager uses the connection tracking features of "netfilter" to route return packets through the required router. When the Traffic Manager receives a new connection, it uses the MAC address in the Ethernet packet header to identify which router the client connected through. The Traffic Manager uses this address to label all packets belonging to the connection with a tracking mark.

When the Traffic Manager sends response packets back to the client, the existence of the tracking mark triggers an IP forwarding rule that sends the packet to the correct router instead of the default gateway or static route.

The Traffic Manager applies Return Path Routing to the following outgoing data packets:

- Packets in TCP connections initiated by a client.
- Packets in auxiliary TCP connections initiated by the Traffic Manager in respect of a TCP connection initiated by a client for certain protocols, such as FTP.
- UDP responses associated with a UDP request from a client.

Note: To ensure correct return path routing of all FTP connections, configure the virtual server handling the traffic to listen on the default FTP port (21).

The following are excluded:

- Other TCP connections initiated by the Traffic Manager.
- Other UDP packets.
- All data handled by virtual servers operating with the SIP (UDP) or SIP (TCP) protocols.

Important: Connection tracking has a significant performance overhead; therefore, you should enable return path routing only if your network infrastructure requires it.

To use this feature, identify the routers in your network infrastructure that you want to use for return path routing. For each, determine the IPv4 (and IPv6, if applicable) address and MAC address of the interface that the IP is raised on. Your router configuration interface might display this information. Alternatively, use network tools such as "ping," "arp," and "tcpdump."

Add this information to the configuration table in the *Return Path Routing* section of the **System > Networking** page of the Traffic Manager Admin UI. This table is replicated to every Traffic Manager instance in your cluster.

To enable Return Path Routing, click the **Yes** radio button and then click **Update**.

Note: You can find errors and warnings due to problems with validation or communication in the event log. However, beyond the initial router connection, the Traffic Manager is unable to detect whether data packets were received through the expected return route. Brocade recommends performing additional network testing to ensure that packets are being received per your requirements.

Configuring IP Forwarding and Network Address Translation (NAT)

Depending on your network topology, the Traffic Manager can function as a network gateway for your back-end nodes or other devices on your network. For example, if you use IP Transparency, you must configure your back-end nodes to route outbound traffic through the Traffic Manager (see *Routing Configuration* in CHAPTER 2).

When the Traffic Manager (or any other network device) acts as a gateway, it forwards traffic from the source nodes to their destination, and forwards responses back to the source nodes.

If all of the traffic forwarded by the Traffic Manager originates from sources with publicly routable IP addresses, there are no complications. All hops in the link between the source and destination know how to route the responses back to the source IP addresses.

However, it is quite common to locate your back-end nodes on a private network that is not routable from the public Internet. With this arrangement, external Internet clients cannot send network traffic to your nodes directly, which conserves public IP addresses and adds a degree of security. A technique called NAT is employed by a gateway device to transparently substitute the unroutable source IP address with the publicly routable IP address of the gateway.

In some network topographies, your downstream router might perform NAT and your internal systems might then all know how to route internal traffic through the Traffic Manager correctly. However, if this is not the case, you configure the Traffic Manager to perform NAT on forwarded packets to ensure that the responses are routed back through the Traffic Manager gateway.

Note: The Traffic Manager's NAT capability is limited to IPv4 only.

To configure NAT, click the **Manage NAT Settings** link in the NAT section of the **System > Networking** page in the Admin UI. Use this page to define rules that govern the behavior of NAT on your Traffic Manager.

In some cases, you might want to use IP Forwarding without configuring NAT: for example, to use the Traffic Manager as a simple IPv4 gateway for your internal network traffic. To do this, tick **Enable IP Forwarding** in the NAT section of the **System > Networking** page in the Admin UI.

Note: This option is unavailable if you have enabled one or more IP Mapping NAT rules or selected any of the Traffic Manager's interfaces to perform NAT. In these circumstances, IP Forwarding is enabled by default.

IP Mapping Rules

The Traffic Manager supports three types of outgoing IP NAT rules. Each replaces the source IP address of the traffic that it handles with the primary IP address of the specified traffic IP group.

These rules are applied in the following order of precedence:

Rule Type	Description
Many to One: Port Locked	NAT is applied to traffic that originates from members of the specified pool, using the specified protocol, and targeting the specified destination port. The protocols valid for this rule type are icmp, sctp, tcp, udp, and udplite.
One to One	NAT is applied to traffic that originates from a single specified IP address. If **Add Inbound Rule?** is ticked, traffic addressed to the primary IP address of the selected Traffic IP Group is forwarded to the specified IP address too, making this rule bidirectional. The source IP address remains unchanged in this case.
Many to One: All Ports	These rules operate similarly to "Many to One: Port Locked" rules, except that they cover all outgoing traffic for the specified pool, regardless of the target port number. Therefore, you can use these as fallback rules for traffic initiated from a pool to any port numbers not covered by your "Many to One: Port Locked" rules.

Port Mapping Rules

The Traffic Manager supports the following inbound NAT rules with your virtual servers:

Rule Type	Description
Virtual Server Port Forwarding	Incoming traffic targeting a specified port range has its destination port replaced with the port that the selected virtual server listens on. This creates the effect of a virtual server listening on a range of ports instead of one. **Note**: Port Forwarding does not interact with outbound IP NAT rules.

Default Rules on Interfaces

Use this section to select an entire Traffic Manager interface on which to perform NAT on the network traffic that is forwarded on behalf of an internal source. Typically, these interfaces have publicly routable IP addresses and are located on the external side of your network (for outgoing traffic).

Note: NAT does not work if the private and public IP addresses are configured on the same NAT-enabled network interface. You must have at least two network interfaces, or more specific rules defined on this page, to create a working NAT configuration.

Enabling this option has the following effect:

- Any traffic that is forwarded by this interface is NAT-enabled: the Traffic Manager replaces the source IP address in the traffic with the IP address of the gateway interface (the Traffic Manager might also replace the source port number with an arbitrary port number).
- From the destination node's perspective, the traffic appears to originate from the Traffic Manager's gateway IP address. The destination node (and all intermediate hops) must know how to route traffic back to that IP address.
- When the destination node replies, the Traffic Manager receives the reply on the interface with NAT enabled. The original source IP address and port is then inserted into the reply and forwarded to the back-end node that originated the connection.

Time and Date Configuration

Note: The settings described in this section will vary slightly for Xen virtual appliance users. Time is synchronized with the host hardware, so the only modifiable setting in this case is the timezone.

It is useful to ensure that the clocks on your Traffic Manager and your back-end nodes are synchronized with each other, so that when you compare log files from each machine, you can determine how events that occurred on each system are related.

If the times are not in sync with each other, it is harder to relate events to each other, such as whether an error logged by the Traffic Manager corresponds to an error recorded by a back-end server.

Use the Time page in the System section of the Admin Interface to manage the time settings on the local Traffic Manager instance.

Note: Unlike most other settings in the Admin Interface, any settings that you configure on this page only apply to the local Traffic Manager whose Admin Interface you are accessing. If you want to inspect or configure the time settings for a different Traffic Manager machine in your cluster, you should use the Admin Interface for that machine.

Setting the Time Manually

Use the Basic Settings section to view the current system time, date and timezone, and to set the settings manually if required.

Using an NTP Server

If you configure all your Traffic Manager instances and your back-end servers to use the same NTP server, this will ensure that their clocks are set accurately and are closely synchronized with each other.

You can use publicly accessible NTP servers, but due to network latency or outages, these may not be sufficiently reliable. You are recommended to run a local NTP service and synchronize all of your local systems to that server. Your local NTP server could itself synchronize from an NTP server managed by your ISP, or a public NTP server.

If for some reason, the time on your Traffic Manager virtual appliance or cloud instance differs from the correct time by a significant amount, for example, 30 minutes, NTP does not adjust for such a large difference. To correct the time difference in this case, click **System > Time** and then click **Sync Time Now**.

Synchronizing Time from the Traffic Manager

Because it typically spans multiple networks, a Traffic Manager appliance/instance may be a suitable, consistent local time source for a variety of networked devices. The Traffic Manager runs a local NTP server that is accessible on all interfaces, so you may wish to configure your other servers to synchronize time from there.

Remote Login to the Traffic Manager

Should you ever need to access the Traffic Manager virtual appliance/cloud instance's command line, you may log in using SSH, and the username and password of a user in the 'admin' group, or a different group with 'Appliance Console' permissions (see the *Permission Groups* section of CHAPTER 24).

Users who log in with appropriate credentials will obtain root access to the appliance/instance; their home directory is `/root`.

If you wish to configure password-less logins using SSH public/private key identification, the SSH configuration directory is `/root/.ssh`.

Entering Custom Kernel Parameters

Sysctl is an interface used to add custom appliance kernel parameters in real time, while the system is running. You can add and modify these parameters through the **System > Sysctl** page of the appliance Admin UI.

Important: This interface is provided for advanced use only. Attempting to change these values could result in unpredictable and unsupported system performance. Brocade recommends modifying these settings only if you are absolutely sure of the results.

Some kernel parameters are controlled by the Traffic Manager and are generated from other configuration settings that you have applied elsewhere. If you attempt to modify one of these parameters, the Traffic Manager responds with an error message and your change is not applied.

Adding or Modifying a Parameter

To add a new custom kernel parameter, enter the parameter name into the *Key* field and its corresponding value(s) into the *Value* field. For multiple values, separate each with a single space. Brocade recommends adding to the optional *Description* field a

description of your parameter for future reference and problem. Click **Submit Entry** to add your parameter to the appliance.

To modify the value of an existing parameter, enter it again with the new value. All changes are reflected in the *Existing Entries* table.

Important: All new and modified parameters take effect immediately.

Existing Entries

To view the description for an existing parameter, place the mouse pointer over the *Info* tag to display it. If no description was entered, the *Info* tag is not present.

To delete custom parameters, check the box next to each one you want to remove and click **Delete selected entries**. All selected parameters are immediately removed; however, you must reboot your server to complete the process. Any default values that were previously overridden are restored.

The Traffic Manager checks the validity of the parameters that you enter. Any observable errors are displayed on this page, in the event log, and on the **Diagnose** page.

Integration with Riverbed Cloud SteelHead

Note: This section applies only to *VMware ESX* and *Amazon EC2* Traffic Manager variants. For more details, refer to your support provider.

The Traffic Manager can be configured to integrate with Riverbed Technology®'s Cloud SteelHead® appliance in order to provide optimized Web services and traffic flow to and from your cloud computing or data center infrastructure.

A client trying to access Web applications hosted in a cloud environment would normally attempt a connection to one of the traffic IP addresses configured on your Traffic Manager instance, which would then perform all usual traffic routing and manipulation according to your traffic management policy for that service. This connection is then usually passed to one of the free back-end nodes that host the service being requested.

A SteelHead can incorporate WAN optimization into this arrangement, although you will require a SteelHead appliance at each end of the connection to facilitate this. Given the circumstances where you are able to route your traffic through a regular physical or virtual SteelHead appliance at the remote client location, you will also

need to configure the Traffic Manager to acknowledge receipt of the optimized traffic and in turn contact a SteelHead in the cloud/data center to complete the connection.

This Cloud SteelHead is effectively inserted into the traffic flow as a proxy to maintain an optimized connection to the client-side SteelHead, which in turn passes traffic to the Traffic Manager in order to balance services to the back-end server nodes as usual.

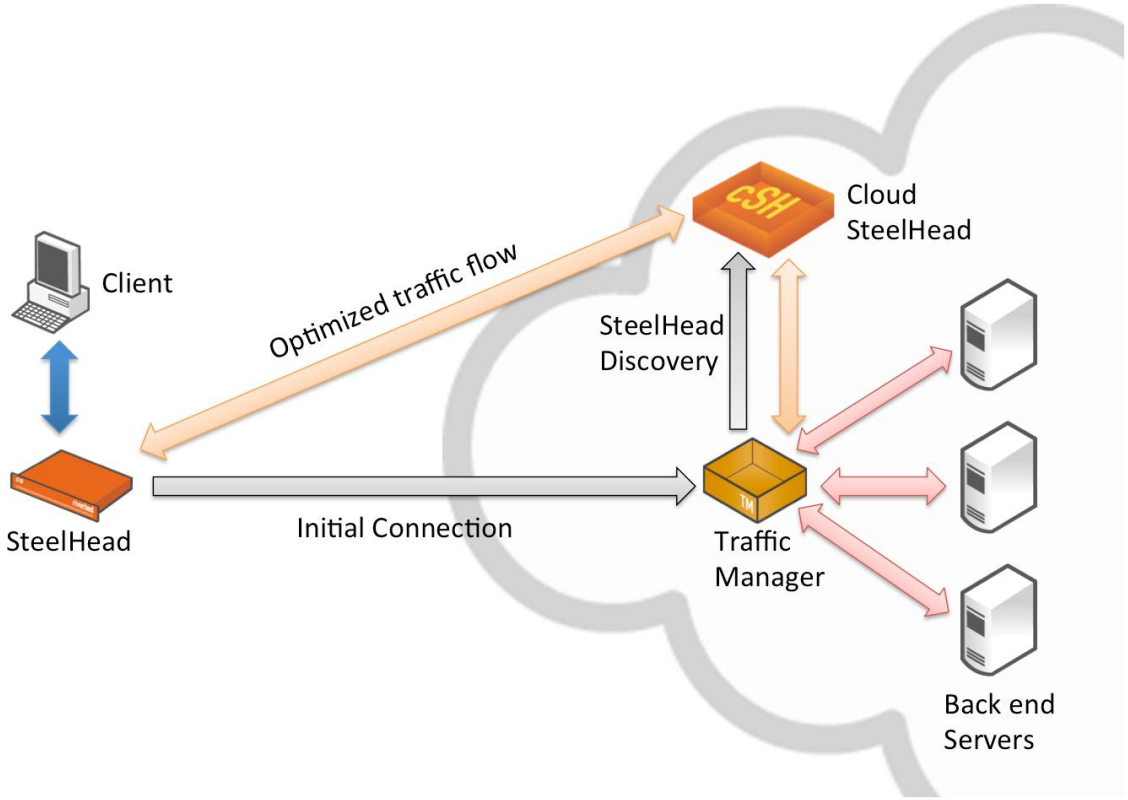


Fig 69. Optimizing the traffic flow to your cloud-based web services

The Traffic Manager provides a feature called the *Riverbed Cloud SteelHead Discovery Agent* that can be used to intercept any traffic identified as optimized and forward it to a Cloud SteelHead in the local data-center or cloud infrastructure. The data is then returned to the Traffic Manager for processing in the normal way. Hence the Cloud SteelHead is inserted into the regular traffic flow between the client-side SteelHead and the Traffic Manager.

The Discovery Agent is controlled from the **System > Riverbed SteelHead** page of the Admin UI. On this page you can provide settings that enable the discovery agent and configure which Cloud SteelHeads this Traffic Manager should use to optimize traffic.

Important: SteelHead Discovery Agent settings are configured on an **individual machine basis**. If you are joining one or more Traffic Managers together to form a cluster, you should ensure that each machine has these settings provided as no automatic configuration replication will be performed for this function.

The available settings are as follows:

Setting	Description
Enable	Setting enabled to **Yes** starts the agent and attempts to connect to a SteelHead using the configured discovery mode.
Mode	The mode used to discover your Cloud SteelHeads in the local data-center or cloud. It can be one of: ▪ **Riverbed Portal:** Use the standard Riverbed cloud portal to manage your Cloud SteelHeads. This is usually only used on Amazon EC2. You must also configure the following settings when using this mode: o **Client ID:** The identifier for this server listed on the Riverbed cloud portal. o **Client Key:** The key for this server listed on the Riverbed cloud portal. o **Proxy Hostname/IP:** If the Riverbed cloud portal is accessed via a proxy, enter its hostname or IP address. Otherwise leave it blank. o **Proxy Port:** If a proxy hostname or IP address is specified you must also enter the port to use. ▪ **Local Portal:** Use an alternate portal to manage your Cloud SteelHeads. You must also configure the following settings when using this mode: o **Portal Hostname/IP**: The hostname or IP address of the portal you are using to manage your Cloud SteelHeads and servers. o **Client ID:** The identifier for this server listed on the portal. o **Client Key:** The key for this server listed on the portal. o **Proxy Hostname/IP:** If the portal is accessed via a proxy, enter its hostname or IP address. Otherwise leave it blank. o **Proxy Port:** If a proxy hostname or IP address is specified

Setting	Description
	you must also enter the port to use. ▪ **Manual:** Manually configure the IP addresses of your Cloud SteelHeads. You must also configure the following settings when using this mode: o **SteelHead IP Addresses:** The IP addresses of all the Cloud SteelHeads you want to use to optimize traffic. If using priority load balancing this list should be in order of priority, with the highest priority SteelHead last. o **Load Balancing Method:** When multiple Cloud SteelHeads are available one is selected using either 'Round Robin' or 'Priority' load balancing. In round robin mode each contactable SteelHead is used in turn. In priority mode the last contactable SteelHead in the IP list is used, continuing up the list.
Log Level	This setting controls how verbose the logs produced by the Cloud SteelHead discovery agent are. The level selected is the lowest severity that will be logged.

Additionally, the **System > Riverbed SteelHead** page provides a Discovery Agent *restart* button should you need to fix an incorrectly performing agent. It also provides a display/download section for recent Discovery Agent logs; the last 30 lines of which are shown (if any are available). These logs can be used to analyze and debug any scenarios where the agent is not performing as you expected.

CHAPTER 23 System Security

This chapter covers important aspects of running your Traffic Manager software in a secure environment. The points in this chapter that consider operating system configuration only apply to the software version, not the virtual appliance or cloud variants.

Firewall and Operating System Settings

The Traffic Manager is not a network-level firewall and it is highly recommended that any Traffic Manager system is placed behind a firewall of your choosing. There are extra measures that can be taken to reduce potential security problems related to your choice of operating system, and UNIX security in general. These are discussed briefly below.

Important: The notes that follow are brief and deal specifically with the Traffic Manager product line. Customers are advised to take extra care over all security considerations related to Internet service providing, and to seek specialized advice if required.

Firewalling Techniques

The Traffic Manager must be able to respond to requests, and certain protocols (such as FTP) may need to open additional connections with clients. Stateful firewalls can be configured to allow outgoing connections to be established from the Traffic Manager on any port, and for subsequent responses from clients to be allowed back to the server.

The recommended firewall configuration for the Traffic Manager is therefore a stateful firewall that denies all inbound traffic by default, allowing named ports on the Traffic Manager to be contacted and any responses to the Traffic Manager to be allowed through the firewall.

Firewall Configuration with the Traffic Manager

The Traffic Manager requires some ports and protocols *in addition* to those used by the virtual servers configured. These are:

1. **The Admin Server port** (HTTPS) - This is usually port **9090** (TCP and UDP), and is the port you use, with the machine name, to browse to the Admin Server interface. This port must be open to any machine from which you wish to access the Admin Server interface.

2. **The control port** - This defaults to **TCP:9080** and is used by the Traffic Manager control protocol to communicate changes in configuration data between Traffic Managers. This port does not need to be exposed outside the cluster. You can check the control port number in *ZEUSHOME*`/zxtm/global.cfg`.
3. **The REST API port** – The default port is **TCP:9070**. The REST API is an alternative programmatic method of configuring the Traffic Manager. For more information, see the *Brocade Virtual Traffic Manager: REST API Guide* available from the Brocade Web site, at http://www.brocade.com.

Both the Admin Server and REST API ports can be redefined as required on the **System > Security** page of the Traffic Manager Admin UI. You should then update your firewall rules accordingly.

The Traffic Manager also uses broadcasts to identify other Traffic Managers. These broadcasts are sent to the multicast address, **239.100.1.1**, port **9090**; all of the Traffic Managers in a cluster listen on this address. To modify the broadcast IP address and port, use the settings in **System > Fault Tolerance**.

If you have configured each Traffic Manager in your cluster to use a dedicated Management Network as described in the *Dedicated Management Network* section of CHAPTER 2, this management traffic will be restricted to that network alone. It should not be possible to connect to any of the Traffic Manager management ports from another network. Nevertheless, you should still firewall these ports off from untrusted sources, for example, in the event that a configuration error relaxes the Traffic Manager's management restrictions, or the management network is accidentally exposed.

Traffic Manager virtual/cloud instances run a local NTP service that listens for NTP (time) requests on all interfaces (see the *Synchronizing Time from* section of CHAPTER 22). The NTP services runs on port 123 (UDP and TCP). It is safe to firewall these ports off if you do not wish to use the Traffic Manager as a local time source.

Important: One of the tests the Traffic Manager uses to detect network availability is ping. The Traffic Manager may not function correctly if your firewall or local TCP/IP tunings disable or rate-limit **ICMP** packets to the default gateway, the back-end nodes and the other Traffic Managers in the cluster.

Network Design

The Traffic Manager is a proxy with advanced traffic management capabilities. This means that the Traffic Manager is normally placed between two networks, relaying requests from a public network to a private network. Each Traffic Manager must be able to contact machines on both the traffic and back-end networks.

For example, the back-end pools may be on an RFC1597 private network, with IP addresses in any of these groups:

```
10.0.0.0/8
172.16.0.0/12
192.168.0.0/16
```

These networks are not routable on the Internet. They offer both convenience in terms of allocating internal network space, and security, since a correctly configured external router will not accept packets for these networks.

It is therefore most secure for the Traffic Managers to have traffic IP addresses (traffic IP groups) that are connected to the Internet, and for all back-end pools to use RFC1597 network addresses on a separate physical network (or correctly configured VLAN). This then prevents back-end servers from being reached directly from the Internet, and allows the Traffic Manager to manage all inbound connections securely and efficiently.

Furthermore, you may choose to use a separate RFC1597 private network for internal management traffic, including access to the Traffic Manager Admin Server. See the *Dedicated Management Network* section of CHAPTER 2 for more details.

UNIX User Permissions

The Traffic Manager software must be started as root. This allows the `zeus.zxtm` process to bind to privileged ports (those below 1024). For example, port 80 (HTTP) is a privileged port. The Traffic Manager also uses superuser privileges to provide fault tolerance; these privileges are needed to raise and lower Ethernet interfaces.

The Traffic Manager processes that manage incoming traffic (i.e. untrusted input) relinquish superuser privileges once they have started. This part of the software runs with the UNIX user and group specified during installation. The Traffic Manager Admin Server (`zeus.admin`) does use superuser privileges to manage configuration that is owned by the root user.

For details about the installation process, see the "Installation and Getting Started Guide" applicable to your product variant. If you are uncertain of the permissions of the Traffic Manager process, check which user owns the `zeus.zxtm` processes using `ps`, or by inspecting *`ZEUSHOME`*`/zxtm/global.cfg`. The user is typically `nobody`, and the group `nobody` or `nogroup`.

To change the permissions, Brocade recommends running the Traffic Manager "configure" script (ZEUSHOME/zxtm/configure) as the system superuser. For more details, see the "Installation and Getting Started Guide".

File System Security

All files within *ZEUSHOME* are configured during installation, and subsequent alterations are made by the Traffic Manager Admin Server. All files within *ZEUSHOME* are therefore owned by the superuser. Certain files have world-readable permissions flags, in order to allow the `zeus.zxtm` processes running with unprivileged permissions to read the Traffic Manager configuration files.

If required, the Traffic Manager can be run within a `chroot` jail, effectively preventing all access to the file system other than *ZEUSHOME* itself. You will need to provide the required system libraries, some device entries in `/dev` and some other system-specific configuration information. For further details, see your system documentation.

Note: Syslog logging is based on UNIX sockets and may still write files outside *ZEUSHOME*.

Operating System Settings

For best security, Traffic Managers should not be used for running applications other than the Traffic Manager, especially applications that provide network services. When a new Traffic Manager is commissioned, pre-installation checks should include removing or disabling UNIX services which will not be used. For example, services such as `lpd` (the UNIX print server) and the RPC Portmapper are often started by default, and are not necessary for the Traffic Manager to function. By uninstalling these services completely, you can dramatically reduce the exposed interfaces that provide a means of accessing your Traffic Manager servers.

Some services may still be required (e.g. Syslog, secure shell). Where practical, these services should be bound only to the private network interface or to the loopback interface if they are not required externally. This will help avoid attracting unnecessary attention to your servers, and should be done even where firewalls are in place.

A good preinstallation starting point is to have only port 22 open on the server, for inbound `ssh` connections, to the management port only (if configured). Open ports can be checked using the freely available `netstat` and `nmap` tools.

Administration of a UNIX server requires regular operating system maintenance. Security is an ongoing process. In particular, it is best practice to track vendor patches and to upgrade services that remain exposed to the Internet as soon as new versions with security-related problems are identified. This applies not only to the services that the Traffic Manager is managing, but also to tools installed on Traffic Managers, such as SSH servers used for administration. For most operating systems,

including Linux, the kernel itself may require upgrades from time to time as security problems are identified and fixed.

CHAPTER 24 Admin Server Security

This chapter describes how to control access to the administration components of the Traffic Manager system – the Web-based administration server, the REST API, the Control API (and, by implication, the CLI) and remote access via SSH (virtual appliance/cloud instances only).

Basic Administration Server Settings

To manage basic security settings, click the **System** button and then the **Security** tab. This will take you to the **Administration Security** page, where you can configure the SSL certificate, IP-based access controls and the ports used by the Administration Server.

Changing the Admin Server SSL Certificate

The first option, **SSL Certificate**, allows you to change the SSL certificate used for the Traffic Manager Admin Server interface. You can choose to generate a new, self-signed certificate, or upload new certificate and private key files. Click **Update** when you have finished.

Note: You cannot choose a certificate from the catalog, since the Admin Server is not a public service but a separate, external process in the Traffic Manager framework.

The SHA-1 fingerprint of the SSL certificate is displayed here. This will be useful for the following:

- To verify the SSL certificate when connecting with a web browser for the first time.
- To verify the authenticity of Traffic Manager identities when joining a cluster.

It is recommended to make a note of the fingerprint upon setting up a new Traffic Manager for the first time. For software variants, it will be displayed on the command line after completion of the installation process. For virtual appliances/cloud instances, you can find it on the console.

Should you need to view the fingerprint again, you can display it from the command line using the following:

```
$ZEUSHOME/admin/bin/cert -f fingerprint -in $ZEUSHOME/admin/etc/admin.public
```

Restricting Access to the Admin Server

As well as using the optional management network, the Traffic Manager can restrict access to the Admin Server using three techniques:

- By client IP address (`xx.xx.xx.xx`)
- By client IP and netmask (`xx.xx.xx.xx/nn` or `xx.xx.xx.xx/xx.xx.xx.xx`)
- By client DNS name or DNS wildcard (`admin.mysite.com` or `*.mysite.com`)

For example, you could specify 10.1.1.1, 10.0.0.0/24 or `*.mysite.com`.

The Traffic Manager must perform DNS lookups if you use DNS rules. You must ensure that the necessary DNS settings have been made to allow these lookups to take place.

Once you have clicked **Update**, the Traffic Manager will check all connections to the Admin Server before allowing access to the login box.

Important: In order for these restrictions to be effective, external firewalls and services must be trusted. IP addresses can be spoofed, and if your DNS service is subverted or hacked then relying on DNS tests may be ineffective.

Changing Admin Server Ports

The Traffic Manager Admin Server is normally run on port 9090 on each Traffic Manager. You can change this port by adjusting the value in the **Admin Server Port** section. Changes take place immediately when you click **Update**. If you change this port, the Traffic Manager will redirect you automatically to the new URL.

This will affect any applications that use the Control API, and the connections made by the CLI. The REST API, however, uses an alternative port (typically 9070) configurable from the **REST API** section.

You can specify a dedicated management port (network interface) at installation time. This value cannot be changed from the user interface. To change this value, or to disable the management port entirely, you need to re-run the configure script used in the initial installation.

For more details, see the "Installation and Getting Started Guide" applicable to your product variant.

Traffic Manager SSH Server Security

Note: This section is not applicable to Traffic Manager software variants.

Traffic Manager virtual appliances and cloud instances provides command line access through a standard SSH server. This is reserved primarily for log retrieval and certain system maintenance operations and is not necessary for normal administrative functions.

Restricting SSH access might form part of an organizational security policy. In this case, enable or disable SSH access for all users by setting **appliance!ssh!enabled** to Yes or No accordingly. Where access is enabled, set the preferred port (usually 22) using **appliance!ssh!port**.

Traffic Manager virtual appliances and cloud instances include an SSH Intrusion Prevention tool to help prevent brute-force SSH attacks. This blocks remote hosts that have made multiple failed connection attempts within a set time period. To enable SSH Intrusion Prevention, set **Enabled** to Yes.

Use the following settings to configure the behavior of the SSH Intrusion Prevention tool:

Setting	Description
Whitelist hosts	A space-separated list of hosts that the SSH Intrusion Prevention tool does not ban (unless the host also appears in the blacklist). Use IP addresses, DNS hostnames, or IP address ranges described by a subnet mask.
Time	The length of time, in seconds, a host is banned for.
Retry limit	The number of failed connection attempts a host can make to the Traffic Manager before being banned. Connection attempts made while a host is banned do not count toward the next ban, unless the time hosts are banned for ("Time") is less than the monitored time span ("Time span"). If this is the case, when the host is unbanned it might still have failed attempts recorded from before the ban started that are within the current monitoring window. Attempted connections while banned are immediately dropped, without an authentication challenge.
Time span	The time window, in seconds, the SSH Intrusion Prevention tool monitors for failed connection attempts.

Setting	Description
Blacklist hosts	A space-separated list of IP addresses or DNS hostnames that the SSH Intrusion Prevention tool must never permit access to this Traffic Manager through SSH. Brocade recommends you ensure that IP addresses and hostnames in the whitelist do not also appear in the blacklist (although the blacklist takes priority). **Note**: Unlike the whitelist, you cannot blacklist an IP range described by a subnet mask.

The Traffic Manager displays currently banned hosts, including blacklisted hosts, under "Banned Hosts".

To download log files containing records of the IP addresses banned and unbanned, the times the SSH Intrusion Prevention tool is started or stopped, and any configuration changes, click **Download Log**.

Cluster Communication

Communication between cluster members is handled via secure communications designed to withstand man-in-the-middle and other types of attacks or interventions. The Traffic Manager provides secure public/private key cryptography using SSL to ensure that your cluster members, regardless of their location in the world, remain able to communicate securely and authentically.

This section allows you to place restrictions on the inter-cluster communications in order to provide the precise level of security for your network setup.

The host IP addresses that can contact the internal administration port on each Traffic Manager can be specified by the setting `controlallow`. This should be a list containing IP addresses, CIDR IP subnets, and `localhost`; or it can be set to `all` to allow any host to connect.

From version 7.0, each Traffic Manager has the concept of being either ‘trusted’ or ‘untrusted’. In the case of clusters that span beyond a single trusted network, it is not always possible to know if a cluster member outside this trusted network has been compromised. In order to prevent other Traffic Managers from becoming compromised, a system has been introduced to enable the administrator to mark certain cluster members as being less secure than others. In other words, if a cluster member is exposed to a higher risk of compromise then it can be marked as such. An untrusted Traffic Manager can still receive configuration updates, and will still broadcast state/statistical data, but effectively becomes unable to replicate configuration updates out to the other cluster members.

To enable this functionality, each Traffic Manager in the cluster has a `control!canupdate` flag. This can either be **enabled** for trusted Traffic Managers, or **disabled** for untrusted ones. You must have more than one Traffic Manager in your cluster to use this setting.

Note: The Admin UI and Control API on untrusted Traffic Managers will be disabled for security purposes. If you need to modify machine-specific settings (e.g. Networking, Time/Date, or SNMP settings) you must do so by first enabling `control!canupdate`. Depending on your network security requirements, it may also be necessary to temporarily remove that Traffic Manager from the cluster in order to make the change.

A default setting `control!canupdate!default` is provided for new Traffic Managers to inherit when they join the cluster. You may wish to modify this to **No** prior to joining a new Traffic Manager from a less trusted location (such as in cloud environments).

Important: As the Admin UI is disabled on untrusted Traffic Managers, you cannot log on to or make changes to those cluster members. Should you run into a situation where your trusted cluster members become uncontactable for some reason, there exists the risk that you may not be able to administer your entire cluster. In addition, you will be unable to join new Traffic Managers in order to regain control of the cluster. In order to circumvent this risk, it is strongly advised that you maintain at least one redundant trusted Traffic Manager in your cluster at all times. Please contact your support provider if you require further assistance.

Important: A compromised pre-7.0 Traffic Manager *may* remain compromised after the upgrade to version 7.0, regardless of the trusted status given to it. This is because the Traffic Manager cannot determine what changes may have been made to circumvent system security prior to the upgrade. It is strongly recommended that you satisfy yourself that a Traffic Manager in an unsecure location has not been compromised prior to upgrading it.

For the same reason, it is also recommended that you do not mix 7.0/pre-7.0 versions of the Traffic Manager software within a cluster spanning an untrusted network. The inter-cluster security improvements in version 7.0 will not become active until all Traffic Managers in the cluster are running the same version.

SSL Settings for Admin Server and Internal Connections

These settings control advanced SSL options for connections to the admin server and secure connections internal to the Traffic Manager. The default settings offer a broad level of security and compatibility suitable for most installations.

Consult your system administrator or support provider should you need to enable compatibility with a specific protocol or connection setting. Brocade recommends taking care when switching between SSL or TLS versions (or cipher suites) due to the potential effect on intra-cluster communication. For example, to switch from TLS 1.1 to TLS 1.2, enable admin!support_tls1_2 and allow your modified configuration to replicate across the entire cluster before disabling admin!support_tls1_1. During this transition, observe the Traffic Manager event log for any persistent warnings or errors that might indicate an incompatibility or communication failure.

Note: If you are using the Traffic Manager REST API and you modify the setting admin!support_tls1_2, you must disable and then reenable the REST API before you can continue to use it with your changed configuration. For information on restarting the REST API, see "Access to the REST API" on page 361.

Access to the REST API

This section contains a number of settings applicable to the Traffic Manager REST API service.

To enable the REST API service, set **rest!enabled** to "Yes". To disable the REST API service, set **rest!enabled** to "No". The service is disabled by default.

To set the TCP port that the service listens on, use **rest!port**. The default is 9070, although any unreserved port can be used provided it does not conflict with other services already running on the Traffic Manager system.

Click **Update** to save any changes.

For information on the remainder of the settings on this page, see the *Brocade Virtual Traffic Manager: REST API Guide*, available from the Brocade Web site (http://www.brocade.com).

User Management

The Admin UI, Control API, REST API and CLI each require a username and password to authenticate each connection. This login authenticates the user and, by way of Permissions Groups, defines the authorization the user has to read, write and otherwise manage configuration and system operation.

In order to configure users and groups, click **System > Users**.

User Authentication

The Administration Server verifies a user's credentials (username and password) against two authentication sources:

- **Local Users -** Credentials are stored in the administration server (as part of the Traffic Manager configuration).

  The Administration Server includes a factory-default local user "admin" that has full administrative permissions (in other words, is a member of the "admin" Permissions Group). In basic administration environments, users log in with this account to perform configuration tasks.

  You cannot rename user accounts, but you can remove them. If you want to remove the "admin" user, you must first ensure that another user account with equivalent admin permissions exists so that you can log in and manage your cluster.

- **Remotely authenticated users -** Credentials are authenticated against externally located systems based on RADIUS, LDAP or TACACS+ services.

  You can configure one or more Authenticators in the User Management page. Authenticators define how the Administration Server verifies usernames and passwords against the database and how it determines the Permissions Group that the user is a member of.

When a user attempts to log in, the Administration Server first compares the credentials to the list of Local Users and determines the Permissions Group that the user is a member of. If a match is not found, the Administration Server then tries each Authenticator in turn until the user is authenticated and a Permissions Groups is determined.

Local Users

To add a new local user, provide a username and password in the **Create New User** box, and select a group from the drop-down box. This specifies which Permission Group this user belongs to.

To modify the password, Permission Group or UI Preferences for an existing user, click the username on the **Local Users** page.

UI Preferences for Local Users

Local Users have access to several preference settings that control aspects of the Admin UI:

- **use_applet**: enables or disables the status applet that is displayed in every UI page. You can disable this applet to reduce bandwidth, or to reduce HTTP requests when using a tool such as LiveHTTPHeaders or FireBug to monitor HTTP traffic to the Traffic Manager IP.

- **appletwidth**: changes the width of the applet, allowing it to display more (or fewer) bars in the chart that graphs traffic per virtual server.
- **trafficscript_editor**: enables or disables the advanced TrafficScript editor, replacing it with a simple textbox when disabled.

To delete a user, click **Delete User**.

Password Policy

The Password Policy page allows you to configure the password policies applied when local users are created or when they change their passwords. Restrictions on the length of passwords, what character types they must contain, how often they can be changed and how often they must be renewed can be configured here.

You can specify the following security settings:

- **No restrictions:** All passwords are allowed, no restrictions are applied.
- **Default restrictions:** Standard password security is applied to new passwords, specifically:
  - Passwords must be at least 8 characters in length.
  - Passwords must contain at least two alphabetic characters.
  - Passwords must contain at least one uppercase character.
  - Passwords must contain at least one numeric character.
  - Passwords must contain at least one special, non-alphanumeric character.
  - Passwords must not contain repeated consecutive characters, such as 'aaaaa'.
- **Custom restrictions:** The minimum length of passwords and the types of characters they can contain can be specified manually.

In addition, the Traffic Manager will maintain a history of passwords used by each user. The setting **password_reuse_after** allows you to specify after how many changes a user can re-use a previous password. This helps to ensure that your users do not simply reset to the same password each time a change is made or required. A value of 0 means a user can re-use any passwords they have previously used.

The setting **password_changes_per_day** specifies how many times a user is allowed to change their password in a 24-hour period. If it is set to 0 then there is no limit to the number of times a password can be changed in one day.

Authenticators

If a user cannot be found in the list of Local Users, the Administration Server will test the credentials against any Authenticators that have been fully configured (and are not disabled). An Authenticator queries an external database, and returns the name of the Permissions Group for users who have valid credentials. Generally the name of the Permissions Group is stored in a field in the remote authentication database, and users who do not have a valid Permissions Group are not allowed access to the Administration Server.

The Administration Server supports LDAP, RADIUS and TACACS+ authentication databases.

Creating an Authenticator

To add a new authenticator, navigate to the **Users > Authenticators** page. Provide a name and select the appropriate authenticator type, then click **Create Authenticator**. This will create an unconfigured authenticator; you will need to provide the appropriate configuration settings for the type you have chosen. Once the authenticator is configured and you have tested it, you can enable it using the 'enabled' setting.

The Administration Server will not use an Authenticator until it has been enabled in this way. If several Authenticators are enabled, the Administration Server will try each of them (in lexicographic order) until one successfully retrieves a Permissions Group.

LDAP Authenticators

LDAP authenticators have the following configurable settings:

Setting	Description
ldap!server	This is the IP address or hostname of the LDAP server.
ldap!port	This is the port used to connect to the LDAP server.
ldap!timeout	The timeout period (in seconds) for a connection to the LDAP server.
ldap!basedn	The base DN (Distinguished Name) for directory searches.

Setting	Description
ldap!filter	A filter that uniquely identifies a user located under the base DN. The string "%u" will be substituted with the username. Examples: "sAMAccountName=%u" (Active Directory) or "uid=%u" (Unix LDAP).
ldap!dnmethod	The bind DN for a user can be constructed from a known string ('Construct') or can be searched for in the directory ('Search' - necessary if you have users under different directory paths).
ldap!binddn	Template to construct the binddn from the username, the string "%u" will be replaced by the username. Examples: "%u@mycompany.local" or "cn=%u, dn=mycompany, dn=local". Only used is the dnmethod is 'Construct'.
ldap!searchdn / ldap!searchpass	The bind DN and password to use when searching the directory for a user's bind DN. You can leave this blank if it is possible to perform the bind DN search using an anonymous bind. These settings are used if the dnmethod is 'Search'.
ldap!groupfilter	If the user record returned by **ldap!filter** above does not contain the required group information you may specify an alternative group search filter here. This will usually be required if you have **Unix/POSIX**-style user records. If multiple records are returned the list of group names will be extracted from all of them. The string `%u` will be replaced by the username. Example: `(&(memberUid=%u)(objectClass=posixGroup))`
ldap!groupattr	The LDAP attribute that gives a user's group. If multiple values are returned by the LDAP server the first valid one will be used.

Setting	Description
ldap!groupfield	The sub-field of the group attribute that gives a user's group. For example, if groupattr is "memberOf" which gives something like "CN=mygroup, OU=groups, OU=users, DC=mycompany, DC=local" you would set groupfield to "CN" (the first matching field will be used).
ldap!fallbackgroup	If groupattr is not defined above, or is not set for the user, the group named here will be used. If not specified, users with no attribute matching groupattr will be denied access.

RADIUS Authenticators

RADIUS authenticators have the following configurable settings:

Setting	Description
radius!server	This is the IP address or hostname of the RADIUS server.
radius!port	This is the port used to connect to the RADIUS server.
radius!timeout	The timeout period (in seconds) for a connection to the RADIUS server.
radius!secret	This is the secret key shared with the RADIUS server.
radius!groupvendor	The RADIUS identifier for the vendor of the RADIUS attribute that specifies an account's group. Leave blank if using a standard attribute * such as Filter-Id.
radius!groupattr	The RADIUS identifier for the attribute that specifies an account's group. May be left blank if **radius!fallbackgroup** is specified.

Setting	Description
radius!fallbackgroup	If no group is found using the vendor and group identifiers, or the group found is not valid, the group specified here will be used.
radius!nas-ip-address	This value is sent to the RADIUS server, if left blank the address of the interfaced used to connect to the server will be used.
radius!nas-identifier	This value is sent to the RADIUS server.

TACACS+ Authenticators

TACACS+ authenticators have the following configurable settings:

Setting	Description
tacacsplus!server	The IP or hostname of the TACACS+ server.
tacacsplus!port	The port to connect to the TACACS+ server on.
tacacsplus!timeout	The timeout period (in seconds) for a connection to the TACACS+ server.
tacacsplus!secret	The secret key shared with the TACACS+ server.
tacacsplus!authtype	The authentication type to use. This can be **PAP** or **ACSII**.
tacacsplus!groupsvc	The TACACS+ "service" that provides each user's group field.
tacacsplus!groupfield	The TACACS+ "service" field that provides each user's group.
tacacsplus!fallbackgroup	If **tacacsplus!groupsvc** is not defined, or no group value is provided for the user by the TACACS+ server, the group specified here will be used. If this is not specified, users with no TACACS+ defined group will be denied access.

Testing an Authenticator

The Edit page lets you test an Authenticator, using the **Test Authenticator** section. You can provide the username and password you wish to test; the Authenticator will run and, if successful, return the name of the Permissions Group that the user is a member of. The Authenticator will also provide detailed debugging information to help you fine-tune or correct your configuration if necessary.

Worked Example: Authenticating Against Active Directory

You can authenticate users against an Active Directory database using the LDAP Authenticator:

Setting	Description
ldap!server	`dir.company.com`
ldap!port	`389`
ldap!basedn	`OU=Company Users, DC=company, DC=local`
ldap!filter	`sAMAccountName=%u`
ldap!dnmethod	`Construct`
ldap!binddn	`%u@company.local`
ldap!groupattr	`memberOf`
ldap!groupfield	`CN`

Test this Authenticator with a username and password:

```
Created LDAP connection to dc2:389
Constructed LDAP user filter: sAMAccountName=username
Using dnmethod: construct
Constructed user binddn: username@company.local
Attempting to bind using supplied password: <hidden>
(length=9)
Searching for user, parameters: filter:
sAMAccountName=username, basedn: OU=Company
Users,DC=company,DC=local
Search returned 1 matches
```

```
Extracted groups using attribute memberOf
   CN=users,OU=Email Lists,OU=Company
Users,DC=company,DC=local
   CN=admin,OU=Email Lists,OU=Company
Users,DC=company,DC=local
   CN=dev,OU=Email Lists,OU=Company Users,DC=company,DC=local
Extracting group using field: CN
Extracted groups:
   users
   admin
   dev
No ldap!fallbackgroup defined
Groups returned by authenticator: users, admin, dev
SUCCEEDED, group: admin
```

Permission Groups

Permission Groups are used to assign access to different parts of the Traffic Manager Admin Server to different users. All local and remote users are members of groups, and these groups define the access users have to the different aspects of the Traffic Manager. By default, Brocade provides a number of pre-defined groups that you can modify or delete according to your local needs. You can also add any number of new customized groups.

To add a new group, provide a name in the "Group name" box and click **Update**. Once the change has been committed, click the new group name.

Each group has a basic "Description" and a "Login Timeout" setting. The description can be used to provide a brief explanation of the purpose of this group, and the login timeout is the period after which inactive Admin Server sessions are terminated. All users that belong to this group inherit this timeout value.

Editing the Permissions of Users in the Group

The Traffic Manager displays a long list of different pages which are available within the Admin Server. All actions are permitted within the ***admin*** group. Your new group starts with all permissions set to **None**.

Each page can have one of the following permissions set for the group in question:

- **None:** Group members cannot use this feature.
- **Read Only:** Group members can view, but not modify items associated with this page.
- **Full:** Members can view and modify items associated with this page.

For example, one of the pages is **Catalog > SSL Certificates**. If you set the permissions to "None", members of this group cannot view the SSL Certificates Catalog page. "Read Only" permissions allow members to view the page but not change settings on it. If you set the permissions to "Full", no restrictions apply.

Note: On some pages (such as catalog summary pages) there are no configurable settings. Blocking access to a page does not restrict access to those hierarchically below it, nor the settings on them.

Important: Groups with access to the **Users** pages can edit their own account details. Brocade recommends taking care when setting who has these privileges.

Login Timeout

The login timeout is measured in minutes, and is the period of inactivity allowed before the Traffic Manager ends your session with the Admin Server. Your session is also closed automatically if you close your browser window, or if you log out manually by clicking **Logout**.

The login timeout is a property of a Permissions Group. Click **System > Users > Permissions Groups** and click the name of the Permission Group you want to modify. Set the new timeout value in "Inactivity Timeout" and click **Update**.

Suspended Users

This page allows you to re-enable one or more users that have been suspended. Users might have been suspended due to exceeding the maximum allowed login attempts, or for some other administrative reason.

Check the box next to each user you want to re-enable, and click **Enable Selected Users**.

Note that individual users can also be re-enabled from their **Edit** page, by setting "Status" to "Active".

Login Security and Behavior

Password policy can be defined on the **System > Users > Local Users > Password Policy** page to control and place restrictions on user login passwords (see the *Local Users* section of CHAPTER 24). In addition to this, you can define further login behavior settings to provide a greater degree of control and security awareness for the users of your system. The **Login and Security Settings** section of the **System >**

Global Settings page provides a number of configuration settings broadly split into two themes:

UI Screen Banners

Should you need to provide your users with a suitable message or reminder upon using the system, you can define these here.

Setting	Description
login_banner	A banner text message to be displayed to anyone who attempts to log in to the Admin UI or Traffic Manager SSH command line.
banner_accept	Whether the user is required to explicitly acknowledge and agree to the `login_banner` text prior to logging into the Admin UI. A check box is added to the login page for this purpose.
uipage_banner	This is a text message that will be displayed at the top and bottom of *each page* of the Admin UI.

Login Controls

You can place controls on the number of login attempts available to users, and the consequences of breaching this.

Setting	Description
max_login_attempts	The number of login failures permitted before a user account is suspended. It will be reactivated after the delay set in `max_login_suspension_time`; however, it can be reactivated sooner from the **System > Users > Local Users > Edit** page or the **System > Users > Suspended Users** page (see the *Suspended Users* section of CHAPTER 24).
max_login_external	Specifies whether externally authenticated (LDAP, RADIUS or TACACS+) users should be suspended after `max_login_attempts` login failures.

Setting	Description
max_login_suspension_time	The length of time for which a user account is suspended after `max_login_attempts` login failures.
login_delay	The delay after a failed login before another login attempt can be made.

The Login Information Banner

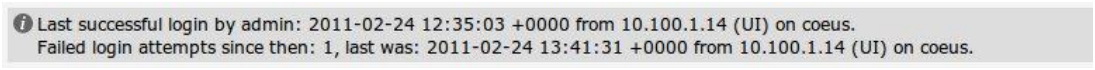

Last successful login by admin: 2011-02-24 12:35:03 +0000 from 10.100.1.14 (UI) on coeus.
Failed login attempts since then: 1, last was: 2011-02-24 13:41:31 +0000 from 10.100.1.14 (UI) on coeus.

Fig 70. Login information

At the top of the Home Page, an information banner describes the previous successful login attempt by the current user on this system. Any previous failed attempts are also shown here, in order that you can identify any unexpected or unauthorized use of your login credentials.

The Event and Audit Logs

The event log contains system and configuration messages, warnings, and errors from all Traffic Managers in your cluster. A truncated version of the event log can be found on the Home page of the Traffic Manager Admin UI. To view the full logs, click **Examine Logs** or **Diagnose > Event Log**.

The audit log monitors recent login attempts, configuration changes (by user and source IP address), and user logouts. It also records changes made using the Control API and REST API. To view the audit log, click **Diagnose > Audit Log** in the Traffic Manager Admin UI.

For virtual appliances and cloud instances, the Traffic Manager controls event log and audit log rotation automatically. Logs are rotated once the maximum file size of 50MB is reached for the current log. As the log is rotated, a new archive file is created for the log entries up to that point, and the current log cleared. Archive files are then compressed with "gzip" and given a filename in the format shown here:

- **For audit logs**: audit-<date>-<sequence>.gz
- **For event logs**: event-<date>-<sequence>.gz

where <sequence> is an incremental number provided for multiple same-day rotations.

For software deployments of the Traffic Manager, you must enact your own rotation policy.

To access archived audit log files, click "Manage archived audit logs" on the **Diagnose > Audit Log** page. To access archived event log files, click "Manage archived event logs" on the **Diagnose > Event Log** page.

For each archived log, you can download or delete the file using the controls provided.

Note: To override the default maximum file size used by the Traffic Manager for audit and event log file rotation, set the system level environment variables `AUDITLOG_ROTATE_THRESHOLD` or `EVENTLOG_ROTATE_THRESHOLD` correspondingly, although certain virtual appliance and cloud variants might have an alternate automatic threshold that is implemented in place of this. For more details, see the "Installation and Getting Started Guide" applicable to your product variant.

CHAPTER 25 The Traffic Manager Control API

This chapter gives a brief overview of the Traffic Manager Control API. For further information, please refer to the *Brocade Virtual Traffic Manager: Control API Guide* included with your distribution.

Information regarding the REST API can be found in the *Brocade Virtual Traffic Manager: REST API Guide*, also included with your distribution.

Introducing the Traffic Manager Control API

The Traffic Manager Control API is a standards-conformant SOAP-based API that can be used to remotely administer and configure a Traffic Manager cluster. The Control API provides an alternative to the Web-based Admin Server and is suitable if you wish to configure the Traffic Manager from another application.

For example, when an Intrusion Detection System detected a remote attack attempt, it could use the Traffic Manager Control API to configure the Traffic Manager cluster to drop all connections from the suspect IP address.

A provisioning system could detect server overloading by monitoring the response times of the server nodes using the Traffic Manager's Service Level Monitoring capability and the SNMP interface. Once it had provisioned additional servers, it could then inform the Traffic Manager by reconfiguring the server pools using the Traffic Manager Control API.

The Traffic Manager Control API can be used by any programming language and application environment that supports SOAP services. Perl, Python, Java and C# are commonly used.

Note: The Traffic Manager Control API is not available on all Traffic Manager product variants. Please refer to your product specifications for full details.

Example: Listing Running Virtual Servers

Perl with SOAP::Lite

```
#!/usr/bin/perl -w

use SOAP::Lite 0.60;

# This is the url of the Traffic Manager admin server
```

```
my $admin_server =
'https://username:password@host:9090';

my $conn = SOAP::Lite
    ->
uri('http://soap.zeus.com/zxtm/1.0/VirtualServer/')
    -> proxy("$admin_server/soap");

# Get a list of virtual servers
my $res = $conn->getVirtualServerNames();
my @names = @{$res->result};

# Establish which are enabled
$res = $conn->getEnabled( \@names );
my @enabled = @{$res->result};

# Print those which are enabled
for( my $i = 0; $i <= $#names; $i++ ) {
   if( $enabled[$i] ) {
      print "$names[$i]\n";
   }
}
```

Run the example as follows:

```
$ ./listVS.pl
Main website
Mail servers
Test site
```

C Sharp or Mono

```
using System;
using System.Net;
using System.IO;
using System.Security.Cryptography.X509Certificates;

public class AllowSelfSignedCerts : IcertificatePolicy {
   public bool CheckValidationResult(
         ServicePoint sp, X509Certificate cert,
         WebRequest request, int problem )
   {
```

```
      return true;
   }
}

public class listVS {

   public static void Main( string [] args )
   {
      System.Net.ServicePointManager.CertificatePolicy =
         new AllowSelfSignedCerts();

      string url= "https://host:9090/soap";
      string username = "username";
      string password = "password";

      try {
         ZXTM.VirtualServer p = new
ZXTM.VirtualServer();
         p.Url = url;
         p.Credentials = new NetworkCredential(
username, password );

         string[] names = p.getVirtualServerNames();
         bool[] enabled = p.getEnabled( names );

         for ( int i = 0; i < names.Length; i++  ) {
            if( enabled[i] ) {
               Console.WriteLine( "{0}", names[i] );
            }
         }
      } catch ( Exception e ) {
         Console.WriteLine( "{0}", e );
      }
   }
}
```

This code works with the .NET 1.1 SDK and with Mono.

Using .Net 1.1, compile and run this example as follows:

```
C:\> wsdl -o:VirtualServer.cs -n:ZXTM VirtualServer.wsdl
C:\> csc /out:listVS.exe VirtualServer.cs listVS.cs
C:\> listVS.exe
```

```
Main website
Mail servers
Test site
```

With Mono, compile and run as follows:

```
$ wsdl -o:VirtualServer.cs -n:ZXTM VirtualServer.wsdl
$ msc /out:listVS.exe /r:System.Web.Services \
        VirtualServer.cs listVS.cs
$ listVS.exe
Main website
Mail servers
Test site
```

Further Examples

For further examples using other programming languages, see the *Brocade Virtual Traffic Manager: Control API Guide*.

CHAPTER 26 Command Line Interface

Your Traffic Manager is normally managed using the Web-based Admin Server. For scripting the configuration, or integrating with other systems, a SOAP-based Control API (CHAPTER 25) and REST API are also available.

However, writing an application to use the Control/REST APIs can take time and may not be appropriate for small, simple tasks or one-off configuration. As an alternative, you can use the command line interface (CLI) for quick access to the full functionality of the Traffic Manager.

The CLI functions as an interactive shell, and can also be scripted. It uses the Control API to communicate with the Traffic Manager cluster, and the commands available in the CLI correspond to the methods available in the Control API.

Accessing the CLI

The CLI is started by running the program `zcli` – which can be found in the following location:

```
ZEUSHOME/zxtm/bin/zcli
```

You can connect to the CLI by logging in to a virtual appliance/cloud instance directly using SSH (see the *Remote Login to* section of CHAPTER 22) and typing the command `zcli`. Alternatively, you can connect remotely by specifying the user and host as arguments to the `zcli` command:

```
zcli [user@]host:port
```

For example:

```
zcli admin@ztm1:9090
```

Usage and options for the `zcli` program can be listed by specifying "`--help`" as an argument:

```
$ zcli --help
Usage: ./zcli [OPTIONS] [user@][host:port] [script file]
Configure the Traffic Manager from a script file or
standard input.

  --user USER        Set the username (default: admin)
  --passfile FILE    Read the password from this file
  --nossl            Your Traffic Manager does not have
```

```
SSL enabled admin server
  --verbose          Verbose output (for testing)
  --continue         When running a script, continue
even if an error occurs
  --help             Show this help
  --version          Show the version of this program
  --formatoutput     Use 'human readable' output, even
in script mode
  --json             Use strict JSON output, even in
interactive mode
  --timeout SECONDS  Set timeout for commands

If no host:port is specified, the local Traffic Manager
will be used.
```

Permissions

The CLI must authenticate itself with the Traffic Manager using a username and password with 'Control API' permissions (see the *Network Design* section of CHAPTER 23). Any user in the 'admin' group has appropriate permissions, and you can add those permissions to other user groups.

By default, the CLI uses the username 'admin' to communicate with the Traffic Manager (the `--user` option can be used to select another username):

```
$ zcli
Please enter your password for user 'admin' on the local
Traffic Manager
Connected to 127.0.0.1:9090
admin@127.0.0.1 >
```

Once connected, the CLI displays a prompt (showing the username and hostname of the Traffic Manager) and waits for a command to be entered:

```
admin@127.0.0.1 >
```

Commands can be typed in one at a time. A command history is maintained, so pressing the up and down arrow keys will step back through previous commands. Auto-completion is also available for many commands; pressing the tab key will auto-complete any half-typed command or object name. Pressing 'tab' twice will list the valid options.

The command help provides a simple overview of the CLI and suggests help topics to find more information on the available commands:

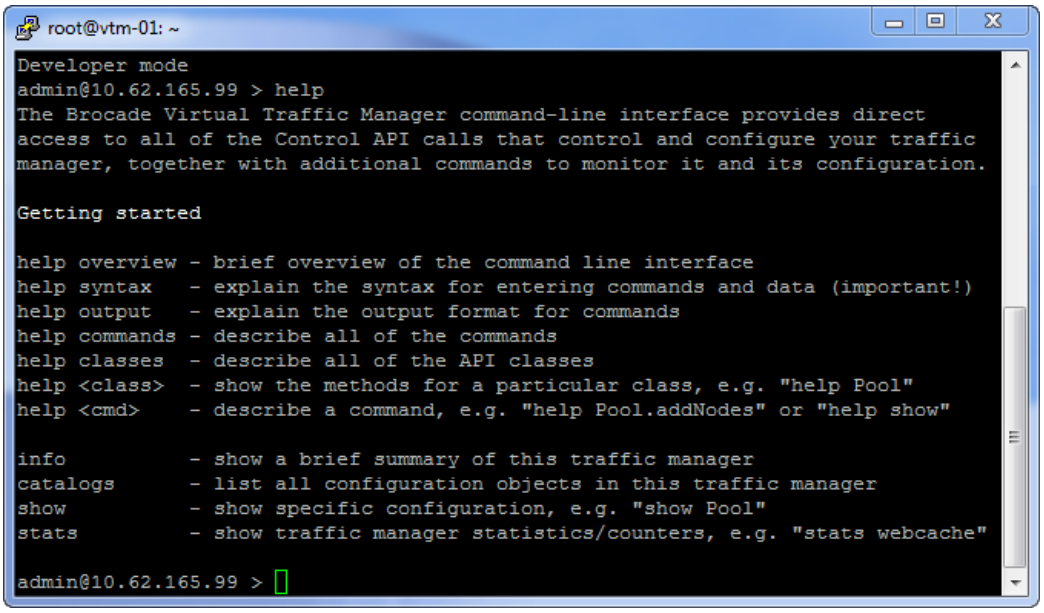

root@vtm-01: ~
Developer mode
admin@10.62.165.99 > help
The Brocade Virtual Traffic Manager command-line interface provides direct
access to all of the Control API calls that control and configure your traffic
manager, together with additional commands to monitor it and its configuration.
Getting started
help overview - brief overview of the command line interface
help syntax - explain the syntax for entering commands and data (important!)
help output - explain the output format for commands
help commands - describe all of the commands
help classes - describe all of the API classes
help <class> - show the methods for a particular class, e.g. "help Pool"
help <cmd> - describe a command, e.g. "help Pool.addNodes" or "help show"
info - show a brief summary of this traffic manager
catalogs - list all configuration objects in this traffic manager
show - show specific configuration, e.g. "show Pool"
stats - show traffic manager statistics/counters, e.g. "stats webcache"
admin@10.62.165.99 >

Fig 71. Top-level CLI help

Commands

To discover the available commands, type `commands` for a detailed list, or use tab-completion (press the tab key twice) for a brief list of commands and objects.

Commands fall into three main groups:

1. **Control API methods**

   These are of the form `Class.method [arguments...]`, for example:

```
admin@127.0.0.1 > Pool.getNodes "Web Servers"
[ "92.52.65.222:80", "92.52.65.213:80" ]
```

2. **Helper commands**

   These summarize the Traffic Manager configuration; the main commands are:

   - `info` - show a brief summary of this Traffic Manager
   - `catalogs` - list all configuration objects in this Traffic Manager
   - `show` - show specific configuration, e.g. `show Pool`
   - `stats` - show Traffic Manager statistics/counters, e.g. `stats webcache`

3. **CLI help**

   - `help syntax` - explain the syntax for entering commands and data
   - `help output` - explain the output format for commands

- `help commands` - describe all of the commands
- `help classes` - describe all of the API classes
- `help <class>` - show the methods for a particular class, e.g. `help Pool`
- `help <cmd>` - describe a command, for example, `help Pool.addNodes` or `help show`

Control API methods

All of the Control API methods are accessible via the CLI. The corresponding command is constructed from the class name and method name as listed in the Traffic Manager Control API manual; refer to this manual for details about each command.

For example, to list all of the virtual servers:

```
admin@127.0.0.1 > VirtualServer.getVirtualServerNames
[ Intranet, "Secure Site", webmail ]
```

Understanding Arguments to CLI Commands

Arguments for commands are given as space separated names; commas between arguments are optional. Argument values that contain a space or other special characters that may confuse the syntax should be "quoted". Lists are contained within '[' and ']' square brackets.

For example, the Control API document for the Traffic Manager details the method 'getNodes' in the Pool class as follows:

> ***getNodes(names) throws ObjectDoesNotExist***
>
> Get the lists of nodes for each of the named pools.

```
String[][] getNodes(
String[] names
)
```

The corresponding CLI command would be `Pool.getNodes`. The documentation shows that the command takes an array of pool names (`String[] names`); the command line to call this method on two pools would be:

```
admin@127.0.0.1 > Pool.getNodes [ pool1 pool2 ]
{ pool1: [ "serverA:80", "serverB:80" ],
  pool2: [ "serverC:80", "serverD:80" ] }
```

The output from `Pool.getNodes` is declared as `String[][]`, i.e. an array of arrays. For each pool that was specified, an array of nodes is returned.

For convenience, in its interactive mode, the CLI indexes the output of the command to show the list of nodes prefixed by the pool name that they refer to.

Using a Single Argument

Many commands in the Control API take a list of arguments so that they can operate on more than one object at once. To simplify the use of commands with a single item, the array can be omitted:

```
admin@127.0.0.1 > Pool.getNodes pool1
[ "serverA:80", "serverB:80" ]
```

Using Wildcards

The '*' symbol is a wildcard that can be used to match multiple objects:

```
admin@127.0.0.1 > VirtualServer.getDefaultPool *
{ Intranet:      pool1,
  "Secure Site": "Secure pool",
  webmail:       "Mail servers" }
```

Partial names can also be wildcarded. For example, to enable keep-alives on all the pools with a name beginning "Pool":

```
admin@127.0.0.1 > Pool.setKeepalive Pool* 1
```

Complex Argument Types

Some commands require more complicated argument structures. For instance, `VirtualServer.addVirtualServer` is defined in the Control API manual as:

```
void addVirtualServer(
   String[] names,
   VirtualServer.BasicInfo[] info )
```

The `BasicInfo` structure is defined as:

```
struct VirtualServer.BasicInfo {
   # The port to listen for incoming connections on.
   Integer port;

   # The protocol that this virtual server handles.
   VirtualServer.Protocol protocol;

   # The default pool that traffic to this virtual
server
   # will go to.
   String default_pool;
}
```

Structures are supplied to the CLI by using '{' and '}' curly brackets, identifying each member with key:value pairs. For example,

```
admin@127.0.0.1 > VirtualServer.addVirtualServer
Intranet { port: 80, protocol: http, default_pool: pool1
}
```

If you need help with the expected arguments for any command, you can use the help available through the CLI:

```
admin@127.0.0.1 > help VirtualServer.addVirtualServer
```

Custom Configuration Sets

You can store and retrieve arbitrary *name:value* configuration pairs in the Traffic Manager configuration system using the CLI. This configuration is replicated across your cluster, and is only accessible through the REST API, SOAP API, and CLI.

You can find a full list of commands associated with storing and retrieving custom configuration by typing `help Custom` at the prompt.

To store a custom configuration, use the `Custom.setStringsLists` command. The following example sets a *name:value* pair under the list named "counters":

```
Custom.setStringLists  [counters], [ {"name":
"Test_Counter", "value" : ["12345", "67890"]} ]
```

You can add multiple *name:value* config pairs to the list by appending additional `{"name":<name>, "value":<value>}` items as desired.

To retrieve the "counters" configuration, use the following command:

```
Custom.getStringLists counters
```

Using this system, you can organize your custom configuration into logical groups, initially by creating a new list, and within this, by each *name:value* pair. Each *value* can itself be a single item or a list of items.

Built-in Commands

The CLI includes a number of additional commands that simplify viewing and modifying configuration. Full help and usage instructions can be found using `help <commandname>`.

- **info:** Shows high-level information about this Traffic Manager, similar to the Home Page of the Admin UI. This shows the Traffic Managers in a cluster, the virtual servers and pools configured, traffic IP addresses, any configuration or operational errors, and the most recent entries in the event logs.
- **connect:** Connect to a different Traffic Manager. The CLI can be used to communicate with remote Traffic Managers, including Traffic Managers that are not in the same cluster as the local machine. This command is also useful when writing scripts, as the Traffic Manager hostname, username and password can all be contained in the script file.
- **errorlog:** Shows the most recent messages in the Traffic Manager event logs.
- **catalogs:** Lists the contents of the configuration catalogs, showing items such as Monitors, TrafficScript rules and Java Extensions.
- **show:** Show the configuration of any configuration object in the Traffic Manager. For example:
  - `show VirtualServer Intranet` - Shows a summary of the virtual server 'Intranet'
  - `show Pool` - Shows a summary of all pools
  - `show VirtualServer customer*` - Shows a summary of the virtual servers customer*

  You can also just use show with an object name, and it will summarize all objects that match that, for example:

  - `show Staging` - Shows a summary of any object called 'Staging'

- `show *` - Shows everything (may take some time)

The **show** command can also be used to provide details about all current connections being handled by the Traffic Manager (by using the command `show connections`). The output is based on the connection info displayed on the **Activity > Connections** page of the Admin UI, with the following data items listed where a connection is in progress:

Data Item	Description
From	Client IP address and port for connection
Via	Local IP address and port the client connected to
To	Back-end node used for the connection
State	The current connection state; can be one of: **C**: Connection to client was closed **f**: Connection finished **R**: Reading from client **W**: Writing to client **X**: Executing rules against client request **c**: Connecting to a back-end node **w**: Writing to a back-end node **r**: Reading from a back-end node **C**: Closing connection with client **K**: Holding client connection in keep-alive state.
VS	The virtual server handling this connection
Rule	The last rule run on the connection
Pool	The pool that is used
BytesIn	Bytes received from the client
BytesOut	Bytes sent to the client

Data Item	Description
Timings	In the format **<num1> / <num2> / <num3>** where: **<num1>**: Time since the client established this connection **<num2>**: Time since last data sent on that client connection **<num3>**: Time since last data sent on the separate back-end connection
Retries	The number of times the server had to retry to connect to a back-end node
SLM	The Service Level Monitoring class being used
VSBW	The bandwidth class the virtual server used
PoolBW	The bandwidth class the pool used
Code	The response code that was returned by the back-end node, varies depending on protocol
Request	Summary of the request made. For HTTP this is the URL requested

Note: Additional fields may also be shown where extra information is identified as applicable to the connection.

The full list of possible object types that can be specified as an argument to `show` is:

```
action, authenticator, bandwidth, ca, certificate,
clientcertificate, connections, event, global, info, java,
licensekey, monitor, persistence, pool, protection, rate,
rule, slm, trafficip, virtualserver
```

- **stats:** Shows counters and statistics for various different objects in the Traffic Manager. For example:
  - `stats VirtualServer Web` - Shows stats for the virtual server 'Web'
  - `stats Pool` - Shows stats for all pools
  - `stats *` - Shows stats for everything

You can also just use stats with an object name, and it will display stats for all objects that match that name, for example:

- `stats Staging` - Shows stats for any object called 'Staging'
- `stats *` - Shows stats for everything (may take some time)

The full list possible stats objects is:

```
authenticator, bandwidth, event, interface, node,
perpoolnode, pool, protection, rate, rule, session, slm,
slmpernode, ssl, system, trafficip, virtualserver, webcache,
zxtm
```

- **help:** Show help topics. For example:
  - `help overview` - Brief overview of the command line interface
  - `help syntax` - Explain the syntax for entering commands and data
  - `help output` - Explain the output format for commands
  - `help commands` - Describe all of the commands
  - `help classes` - Describe all of the API classes
  - `help <class>` - Show the methods for a particular class, e.g. `help Pool`
  - `help <cmd>` - Describe a command, for example, `help Pool.addNodes` or `help show`
- **rule:** Upload, download, and syntax check TrafficScript rules on the Traffic Manager.
- **getbackup:** Download a backup from the Traffic Manager.
- **putbackup:** Upload a backup file to the Traffic Manager.
- **download:** Download a specific configuration file from the Traffic Manager, for offline editing or archiving.
- **upload:** Upload a specific configuration file to the Traffic Manager.
- **edit:** Download and edit a specific configuration file from the Traffic Manager, and then upload the modified version.
- **watch:** Repeatedly run a command and watch the output. This is useful for monitoring changes, e.g. keeping an eye on logs or recording stats. For example:

```
admin@127.0.0.1 > watch 10 stats webcache
```

This will show all of the Web cache statistics, updated every ten seconds.

Scripting the CLI

The CLI normally runs in 'interactive' mode, where commands can be entered from the keyboard and are run instantly.

It is also possible to run a script of CLI commands without user input. For example, the following script prints out the Web cache statistics, clears some items from the cache and then prints the Web cache statistics again:

```
System.Stats.getWebCacheEntries
System.Cache.clearCacheContentItems "Intranet" "http"
"www.example.com" "*/product/*"
System.Stats.getWebCacheEntries
```

Save the script to a file on the local machine and then execute it as follows:

```
$ ZEUSHOME/zxtm/bin/zcli /home/user/webcache-script
```

The CLI will read the file and run each command found in turn. Providing the script completes successfully, an exit code of 0 (zero) is returned. If an error occurs when running a command, script execution will stop and the CLI will exit with a non-zero return value. However, where `--continue` is specified at the command line, the script will continue regardless of any errors encountered during processing. In this case the return value is still 0, except where the CLI encounters an internal error of some kind (such as the use of incorrect parameters).

To fully automate a script, the username and password must be specified. This can be achieved in two ways:

1. Store the password in a file and provide this file to the `zcli` script:

   ```
   ZEUSHOME/zxtm/bin/zcli --user username --passfile
   /home/user/password /home/user/webcache-script
   ```

   The file containing the password should have suitably restricted permissions so that other users cannot read it.

2. Add the `connect` command to the start of the script, so it becomes:

   ```
   connect localhost 9090 username password
   System.Stats.getWebCacheEntries
   System.Cache.clearCacheContentItems "Intranet" "http"
   "www.example.com" "*/product/*"
   ```

```
System.Stats.getWebCacheEntries
```

Script Output

The output from scripted CLI commands is slightly different from interactive CLI commands. In interactive mode, the output is formatted and spaced to make it more human-readable, spreading the output over several lines if needed. In scripted mode, the output from each command is written on one line only, without any extra padding or formatting.

The output format is JSON-formatted, to facilitate importing and parsing in other programs. Output from commands that take several inputs (e.g. multiple pool names) is not presented in a table to highlight the results for each item.

For example, compare the output from the same command run on interactive and scripted sessions.

Interactively:

```
admin@127.0.0.1 > Pool.getNodes [ pool1 pool2 ]
{ pool1: [ "serverA:80", "serverB:80" ],
  pool2: [ "serverC:80", "serverD:80" ] }
```

From within a script:

```
$ ZEUSHOME/zxtm/bin/zcli --user username --passfile
/home/user/password
/home/user/getnodes
[["serverA:80","serverB:80"],["serverC:80","serverD:80"]
]
```

If you want to keep the human-readable output, then run `zcli` with the `--formatoutput` argument.

CHAPTER 27 Granular Configuration Import/Export with zconf

Introduction

If you are likely to provision a significant number of Traffic Managers on a regular basis, any way to minimize duplication of effort during the provisioning process is a way to decrease time-to-deployment costs. Quite often, users will have a standard set of policies or common pieces of configuration that they want to roll out to a large number of services. From time to time, one may wish to make the same configuration change across a large enough number of Traffic Managers.

Users may have some kind of change control database to store policies or configuration files. Subversion (SVN) is an example of a tool that can be used to store and manage changes to not only source code, but configuration files as well. The Traffic Manager includes a tool called `zconf` that can be used to perform granular levels of configuration export and import and to store those [partial] configurations in a tarball. If desired, the tarball can be extracted and checked into one of these change control databases.

Using zconf

The `zconf` utility can be found in `$ZEUSHOME/zxtm/bin`, where `$ZEUSHOME` is the directory that you installed the Traffic Manager to. Note that if `$ZEUSHOME` is not set correctly that the `zconf` utility will not function correctly. This command line utility has a built-in help system. When you run the program with no arguments, you should see the following output (this example taken from the Traffic Manager virtual appliance):

```
root@ubuntu1:~# $ZEUSHOME/zxtm/bin/zconf

ERROR: No command specified.

Usage: /opt/zeus/zxtm/bin/zconf [COMMAND] [OPTIONS]
[ARGUMENTS]
Commands:

  copy
  diff
  export
  getdefaultfilter
  help
```

```
  import
  info
  list
  replicate
  version

For more information on a given comment execute:

 /opt/zeus/zxtm/bin/zconf help <COMMAND>
```

This will show you a list of commands. To learn how to utility works, you should read the help for each command.

Exporting a Complete Backup

If you want to take a complete backup of your Traffic Manager's cluster configuration, use `zconf`'s export command. By default, the output of the export command is printed to STDOUT. If you want to save the output to a file, you can use either the `--file` argument, or redirect the output from STDOUT to a file (as in the example below). On most Linux or Solaris systems, you can then use the `tar` command to extract the contents of the tarball (for example, for checking in to a subversion repository).

```
root@ubuntu1:~# $ZEUSHOME/zxtm/bin/zconf export >
backup.tar
root@ubuntu1:~# ls
backup.tar
root@ubuntu1:~# tar xf backup.tar
root@ubuntu1:~# ls
backup.tar  conf
root@ubuntu1:~# ls conf
actions          groups         monitors  security
users
activitymonitor  licensekeys    PARTIAL   services
VERSION_8.0
commkey          locations      rules     settings.cfg
zxtms
events           locations.cfg  scripts   TIMESTAMP
root@ubuntu1:~#
```

Configuration Listings

Configuration backups are stored in a number of plain-text files and organized into a directory tree. The `list` command of `zconf` can be used to list the configuration files in the current running Traffic Manager and any configuration backup or backup archive. Say for example that you want to get a list of the virtual servers configured on the current Traffic Manager:

```
root@ubuntu1:~# $ZEUSHOME/zxtm/bin/zconf list vservers
root@ubuntu1:~#
```

This output indicates that the `vservers` directory is empty, and consequently that there are no configured virtual servers. Suppose you want to list the virtual servers in a configuration archive:

```
root@ubuntu1:~# $ZEUSHOME/zxtm/bin/zconf list vservers
                     --conf-path backup.tar
monitors/Full HTTP
pools/Bar
vservers/Foo
```

Notice that while you specified `vservers` as the glob argument to the `list` command, that a pool and a monitor are also listed in the output. This is because all dependencies required to make that virtual server work as expected are printed.

Partial Imports

Assuming that you have a configuration archive tarball in the current directory, in order to import a virtual server (called "Foo" for example) and all of its dependencies you should use the following command:

```
zconf import --include vservers/Foo backup.tar
```

This import will cause the configuration to be replicated across the other members of the cluster.

Note: Similar functionality is also available from within the Admin UI, on the **System > Backups > Partial Backups** page. Please refer to the *System > Backups > Partial* section of CHAPTER 7 for more details.

CHAPTER 28 Multi-Site Cluster Management

Introduction

The multi-site cluster management functionality discussed in this chapter builds on the sophisticated capabilities of the Traffic Manager by adding support for management of *multiple* distributed physical, virtual or cloud-based Traffic Manager clusters. These clusters could be present at various geographic locations, on various platforms and include varying numbers of Traffic Manager instances. Each might host specific services that you want to administer from one central location.

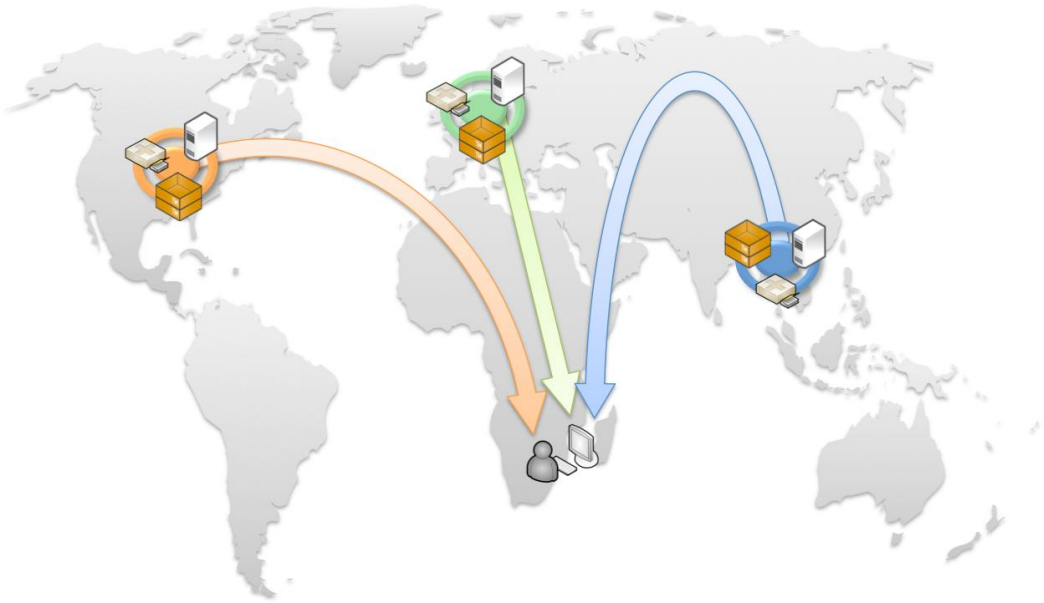


Fig 72. Centrally managing your global Traffic Manager cluster infrastructure

This chapter discusses how your globally distributed Traffic Manager infrastructure can be pulled together into one centrally managed cluster - while retaining the location-specific configuration of your Traffic Manager systems.

This feature is not to be confused with Global Server Load Balancing (GSLB), which involves geographically aware load balancing across services hosted at multiple locations around the world. The implementation of GSLB in the Traffic Manager is discussed in CHAPTER 29 below.

This chapter assumes you are familiar with the Traffic Manager user interface and general configuration concepts described elsewhere in this manual. It is strongly recommended that you refer to CHAPTER 7 before continuing.

Activation and Deactivation

You can enable multi-site management functionality from the **System > Traffic Managers** page of the Admin UI. Simply tick the **Confirm** checkbox and then click **Enable** in the *Multi-Site Management* section to activate the Traffic Manager in this mode.

Equally, the Traffic Manager can be set back to its original state by deactivating the multi-site capability in the same way. In this scenario, you must first select one of your defined configuration locations as the *master* configuration to be adopted by all Traffic Managers. All other configuration locations are then removed along with any location-specific configuration values stored for them. Ensure your chosen location is selected in the drop-down list, tick the **Confirm** checkbox and then click **Disable**.

Note: Although configuration locations are removed, GLB locations remain - but in a deactivated state.

Key Concepts

Configuration Locations

Configuration locations are essentially groupings of Traffic Managers, perhaps situated in a particular physical location, that are required to host the same services within an overall cluster. Configuration locations are defined within the **Catalogs > Locations** page, and each of your Traffic Managers is then marked as present at a particular one.

Throughout the Admin UI, services and other configuration aspects can be made location-specific, and only those Traffic Managers marked as present at that location will use them.

This concept provides the ability to combine all of your globally located Traffic Manager application delivery requirements into one centrally administered system. From a single Admin UI, you can now configure and monitor all your services regardless of the complexity of the setup.

Note: It is not mandatory to have your Traffic Managers physically present at a particular location. The concept of a configuration location, as opposed to a GLB location, is mainly as a mechanism to allow virtual grouping within your Traffic Manager cluster.

Clusters

Within a multi-site configuration, a cluster is a group of Traffic Managers deployed over one or more global locations, centrally managed, in order to provide the delivery of application services in a fault-tolerant manner.

Traffic Manager service configuration is replicated between all machines in the cluster. As with non-multi-site environments, the majority of the configuration you make is shared, however multi-site management extends this facility by providing the ability to set location-specific configuration that is only active on those Traffic Managers marked at that location. This provides a form of sub-clustering, or local service delivery. You can edit the overall configuration using the Admin UI of any Traffic Manager in your cluster and it will be replicated to all other Traffic Managers.

A small amount of system configuration is specific to each location (such as local DNS servers), and a small amount of system configuration is specific to a particular machine (such as SNMP settings). To configure location-specific system information, you can use the UI of one of the Traffic Managers in that location, and it is replicated to other Traffic Managers defined at the same location. To configure machine-specific system information, you must use the UI on that machine.

Important: Where you are intending to join a remotely located Traffic Manager into a cluster, it is essential that you provide it an **External IP address** first. Furthermore, it is recommended that you ensure the availability of this address, and in particular ports 9080 and 9090, across your cluster prior to adding the new machine. Failure to observe this could cause configuration replication problems within your cluster. See "Setting Traffic Manager Locations" on page 398 for more details about how to set the External IP.

Deployment Scenarios

Note: All references to 'location' in this section refer to **Configuration Locations**. Additionally, all scenarios assume suitable IP/port connectivity to external locations (see above).

Create and Manage a Second Traffic Manager Location

As your services grow in popularity and sophistication, you may wish to deploy a second Traffic Manager presence in some other global location to be run concurrently.

- First, perform installation and any initial configuration required by each of the new Traffic Managers at the new location. Please note however that when joining

an existing Traffic Manager cluster, any non-system specific configuration will be overwritten as the cluster config is replicated to each new machine being added.

- From the Admin UI of each Traffic Manager in turn, run the **Join a Cluster** wizard to connect to your existing Traffic Manager cluster.
- Either pre-define the new second configuration location on your existing cluster, or specify the 'new location' option on step 4 of the wizard. Note that specifying a new location during the wizard will not apply a value to the *position* setting in the location definition. You will need to do this manually.
- For each subsequent Traffic Manager, while running the wizard you should ensure you select the newly created location on step 4.

Each new Traffic Manager will now be added into the cluster, with a copy of the full configuration, but marked as present at the second configuration location. Any configuration changes that were, or are subsequently, made in a location-specific way will be applied across the whole cluster.

Add a New Traffic Manager to Your Multi-Site Cluster

You can add additional Traffic Managers to any location managed by the Traffic Manager by running the **Join a Cluster** wizard from the Admin UI of the machine you are adding. This will import the configuration of your cluster to your new Traffic Manager and it will appear as a managed machine within the Admin UI.

Merging Two or More Existing Traffic Manager Clusters

In normal cluster joining operations, the joining machine takes on the configuration of your cluster, thus deleting any existing configuration you might have had. What we are attempting to do here is to merge two or more distinct configurations so special care must be taken to preserve the settings on all machines.

The Traffic Manager provides a command line tool 'zconf' to perform fine-grained config import/export operations. With this tool we can choose to import the individual services of the secondary clusters into the master cluster config without compromising the existing services. It is also worth making use of *zconf*'s built-in "dry-run" mode for any import procedures, to ensure that you are fully aware of the expected outcome. For full usage syntax and further information, please refer to the Online Help available directly from *zconf*.

The intention is to minimize service downtime as best as possible, however it is important to note that there may be service interruptions as Traffic IP addresses are transferred so this is best performed at a time where your Web traffic is at its least.

In all cases, it is strongly recommended that you back up the configuration on all clusters before attempting this procedure. It is also worth attempting the

following with a non-critical system first, in order to satisfy yourself of the likely outcome.

Perform the following steps:

- Ensure that all your Traffic Managers in all clusters are running the latest Traffic Manager software. Designate one cluster as the *master* cluster.
- Set up any new secondary configuration locations on your master cluster.
- In order to minimize filename clashes, you may wish to rename your virtual servers, rules, pools, etc., on all clusters to uniquely identify them, such as by pre-pending the location name. For example, renaming "Intranet" to "london-Intranet" and so on. This is not mandatory, however it should reduce the chances of config clashes. The default behavior of *zconf* is to overwrite existing same-name config files when you perform an import, however this is switchable.
- Export your remote cluster configuration using the **System > Backups** page, and copy the resultant tarball over to your master cluster.
- Using *zconf*, import the virtual servers (`/vservers`) directory from your config backup. *zconf* will automatically follow dependencies in each virtual server file and import in the relevant supporting files. The only caveat with this is the use of any Java extensions - you may need to manually import in the `/jars` and `/servlets` directories yourself.
- These imported services should now be marked as enabled only for the newly defined remote location. This can be achieved on the **Virtual Server > Edit** page by clicking the *Edit setting by location* icon next to the *Enabled:* setting. Ensure this virtual server is enabled for the remote location and disabled for all other locations.
- Use of Traffic IP Groups can cause service conflicts if you have the same IP addresses set up on two clusters. You may need to take the IP addresses in your existing remote cluster offline in order to set up the same IP addresses in your master cluster. As you add in your remote Traffic Managers to the master cluster, you will need to first remove them from the remote cluster, and as a result, any remote IP groups they appear in first.
- Assuming you have multiple Traffic Manager machines in each cluster, you can disconnect a single Traffic Manager from each and join it to the master cluster, marked as being at the corresponding defined remote location. **Please note that system specific settings should be applied directly through the Admin UI of the machine they pertain to.**

- This Traffic Manager can be added to the newly created Traffic IP Group and your services should resume.
- Now you can take each remaining remote Traffic Manager out of the remote cluster and join them into the master cluster, marked as being at the appropriate location and added to the correct Traffic IP Group. Each will inherit the fully merged configuration, and yet will only respond to the services marked for that location.

Configuration

Setting Up Locations

Locations are listed and modified from the **Catalog > Locations** page. You can also find them through the **System > Traffic Managers** page, by clicking on the *Manage Locations* link. Your defined locations are listed here, with the opportunity to add to, and remove from, the list.

Note: For product variants that include **Global Load Balancing**, a separate additional section is shown for "GLB Locations". Please see CHAPTER 29 below for more details.

Each location is simply defined by its name. Clicking on the location name or associated **Edit** link allows you to modify its settings or delete the location altogether.

Adding a new location can be achieved using the *Add new Location* section under the main list. All new locations are created using an existing location as a template (excluding the case where there are no existing locations). **This 'Based on' mechanism copies any location-specific config for the template location to your new location.**

Locations can be deleted from the section at the bottom of the associated Edit page. Note however that locations in use by one or more Traffic Managers cannot be renamed or deleted. You must first disassociate your Traffic Managers with this location before such operations can be performed.

Setting Traffic Manager Locations

On the **System > Traffic Managers** page, each Traffic Manager listed in your cluster has two new configurable settings:

- **Location:** Select this Traffic Manager's configuration location from the drop-down list.

- **External IP:** When your Traffic Manager's host name is not globally resolvable, you will need to specify an externally available IP address for communications between your cluster members.

Any changes to these settings must first be confirmed by clicking the checkbox next to the **Update** button.

Location-Specific Configuration

A significant change to the way the Traffic Manager handles configuration settings while in multi-site mode can be seen by the addition of a new icon next to each individual setting.

Fig 73. The Location icon

Clicking this icon separates out the setting into multiple instances, one for each configuration location. Clicking the icon again will revert the setting back to being a singular instance - which is used by all locations.

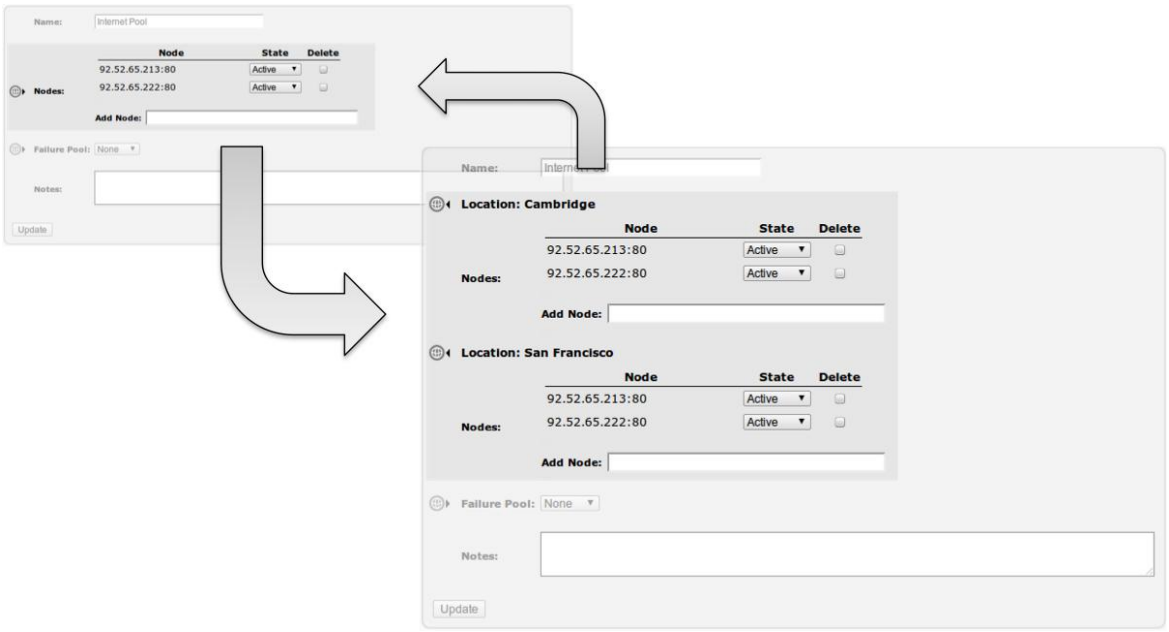


Fig 74. Switching your node list between location and normal mode

This mechanism allows you to specify individual configuration settings on a per-location basis, while maintaining some global settings that apply to all cluster members.

Home Page Changes

When multi-site cluster functionality is enabled, a modified admin UI home page is presented. The new style is more suited to the environment within which the Traffic Manager is expected to operate. It now shows a number of distinct sections as listed below:

- **Locations:** Your Traffic Manager configuration locations are shown here. Locations defined but not in use will not be shown. Click the location name to view the **Catalogs > Locations > Edit** page for that location.
- **Services:** This shows the list of services marked as *enabled* and managed by this Traffic Manager installation. A service can be considered as a combination of virtual server, rules, pools and other associated objects. Click the service name to view the **Virtual Servers > Edit** page for that service.
- **Event Log:** This is a quick summary of the event log. It is written-to by all Traffic Managers with the name of the machine in question alongside each entry. The full logs can be viewed by clicking the *Examine Logs* link.

Each of the locations and services are displayed with a suitable status indicator:

Indicates running ok

Indicates a warning

Indicates a serious error

Fig 75. Status indicator icons

Warnings are non-critical problems that mean the location or service is still running, but with impaired performance. Errors require action to rectify a problem that is preventing your locations or services from operating. The status applet will reflect these conditions. In the case of warnings or errors, clicking on the location/service name takes you to the **Diagnose** page, where you will see a full explanation of the problem.

The World Map

Fig 76. The world map

Click **Activity > Map** to show a world map for enhanced visualization of your locations and services. You can also view the map via the *View traffic on World Map* link from the **Virtual Servers > Edit** page. This provides a global overview of the locations defined on your system.

The map is designed to be interactive and tools are provided to move around and zoom into a level of detail required for your infrastructure. You can center the map on a geographic location of your choice by holding down the left-hand mouse button and dragging the map to your required destination. You can show traffic for a specific virtual server using the drop-down control at the bottom. The default is for *all* virtual servers.

Each colored target represents a location, and each small dot that appears on the map represents a user who has issued a request for a particular service. Each user is colored according to the location at which the requested service is hosted. In order to show meaningful information, you will need to enable Connection Archiving (also required for viewing advanced connection analytics). If this is currently disabled, you can click the *connection archiving is disabled* link at the bottom to enable it. This link takes you to the **System > Global Settings** page where you will need to provide a suitable value, e.g. 500, for the **recent_conns** setting (the default of 0 disables connection archiving).

Traffic Visualization

As with Global Load Balancing, activating multi-cluster management functionality on your system automatically enables additional options into the Traffic Manager's existing diagnostic and visualization tools. Please refer to the *Traffic Visualization* section of CHAPTER 29 for more details.

CHAPTER 29 The Traffic Manager DNS Server

The Traffic Manager provides a built-in authoritative Domain Name System (DNS) server capability. This enables the Traffic Manager to host zone files and answer DNS queries.

For Global Load Balancing (GLB) deployments, you can elect to employ the internal Traffic Manager DNS Server or balance traffic to a pool of external DNS Server nodes.

For an introduction to DNS terminology, see "DNS Primer" on page 402. To configure the Traffic Manager as a DNS server, see "Configuring the DNS Server" on page 411.

DNS Primer

Introduction

The DNS can be considered the phonebook of the Internet. DNS is used to look up different bits of information about domain names. For instance, your Web browser needs to know which IP address to connect to in order to request a Web site from a Web server. Similarly, when you send an email to someone, your email server needs to know both the domain name and the IP address of the mail server responsible for delivering the message to its recipient. DNS provides the means with which to find this information.

The Layout of DNS

The layout of DNS resembles a top down tree structure. The root of the structure consists of a group of *Root Name Servers*. Below the root are the *Top Level Domain Servers*. The names "com" (as in "www.example.com") and "net" are both examples of top level domains. Below the top level, DNS provides no specific classification beyond the term "Name Server".

Delegation of Authority

"Authority" in the context of DNS refers to the responsibility of a name server to handle requests for a given domain. If a name server is *Authoritative* for servicing DNS requests for anything under example.com, it has been delegated the authority to do so by the higher-level domain server(s). The Traffic Manager's built-in DNS capability is an example of an authoritative DNS server.

Name Resolution

The process of "looking up a piece of information" using DNS is called "resolution". For example, when your Web browser needs to find an IP address for a Web site, it attempts to resolve the domain name to get an IP address. When your email server needs to know which mail server is responsible for delivering a message to a particular domain, it attempts to resolve the "Mail Exchange (MX)" name and its IP address.

As part of its network configuration settings, a computer typically knows of at least one "Name Server". That Name Server in turn provides the computer with responses to DNS requests. Only under special circumstances[11] does a computer send DNS requests to a different Name Server.

Resource Records

Each piece of information for a domain name is held in what is called a Resource Record. Typically, Resource Records are distinguished by type. For instance, when your Web browser wants to know the IP address for www.example.com, it sends a query for an "A record" that matches www.example.com. If your email server needs to know where it should send a message addressed to john.smith@example.com, it sends a query for an "MX record" that matches example.com. Some commonly used and supported resource record types are:

- **SOA (Start of Authority) -** Provides a few pieces of useful information about a given domain.
- **NS (Name Server) -** Indicates the domain name of a name server that is authoritative for a given domain. This resource determines how authority is delegated from a higher-level domain to a lower-level domain.
- **MX (Mail eXchange) -** Indicates the domain name of a mail server that should accept mail for a particular domain.
- **A (Address) -** Contains an IPv4 address of a domain name.
- **AAAA (Address) -** Contains an IPv6 address of a domain name.
- **CNAME (Canonical Name) -** Contains an alias for a domain name.
- **PTR (Domain Pointer) -** Used for reverse lookups, from IP back to DNS name.
- **SRV (Service Record) -** Defines the hostname and port number of servers used for specified services.

[11] As a practical example this would be at the hands of a DNS administrator who is performing testing.

- **TXT (Text Message) -** Contains an arbitrary human readable text message, used for many purposes.

Zone Files

Resource Records are typically stored in text files on authoritative DNS servers[12]. A zone file contains a set of records that are specific to a domain. Each zone file must contain at least an SOA record and an NS record.

The Traffic Manager supports zone files with the following example format:

```
$TTL    604800   ; This TTL declaration applies to all
                 ; records defined below that do not define
                 ; their own TTL

; In this file, @ means the "current origin domain"

@       IN      SOA     ns.example.com. root.example.com. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800         ; Negative Cache TTL
                        )

@               NS      ns
ns              A       192.168.0.1
@               NS      ns.example.com.
ns              A       192.168.0.254

@       IN      MX      1 host
@       IN      MX      2 mail2.example.net. ; note this is
                                             ; in another
                                             ; zone/domain

host    IN      A        192.168.0.2
                AAAA     2001:db8:85a3:0:0:8a2e:370:7334

$TTL 1w2d3h4m5s  ; The TTL defined here overwrites the
                 ; setting above and applies to all the
                 ; records below it that do not define their
```

[12] Some DNS implementations can instead store resource records in a database; however, the Traffic Manager DNS server uses the zone file method described here.

```
                ; own TTLs.

foo.bar             A     192.168.0.3 ; An example of a
                                      ; relative address:
                                      ; foo.bar.example.com.
foo.bar.example.com. A    192.168.0.4 ; An example of an
                                      ; absolute address

alias               CNAME  host

_sip._udp           SRV    0 5 5060 sipserver
sipserver    3600   A      192.168.0.5  ; This record has
                                         ; its own TTL (3600)

text1             TXT   "This is a TXT record"
*.wildcards       TXT   "This is a wildcard TXT record"
```

The Traffic Manager parses zone files using a top-down approach, and it assumes inheritance from the preceding line where indentation occurs.

If you modify or update your zone file after the initial deployment, you must increment the "serial" counter to ensure the DNS service is made aware of the change.

A zone file typically contains at least one TTL (Time To Live) definition, in particular at the top of the file. The Traffic Manager uses this value to determine how long clients should wait before refreshing DNS-related information. The following rules and observations apply to TTL values in zone files:

- If you do not define a TTL value in your zone file, the Traffic Manager uses instead the Negative Cache TTL value from the SOA definition. The Negative Cache TTL is normally used as a TTL in "negative responses". That is, NXDOMAIN (domain name not found) and NODATA (domain name found, but no records of given type).
- All records with the same domain name should have the same TTL value.
- A TTL can be specified as a single number of seconds or the alternative format of 1w2d3h4m5s to mean 1 week, 2 days, 3 hours, 4 minutes, and 5 seconds. The alpha components of this format are case insensitive, and they can be specified as uppercase, lowercase, or mixed (for example, 3W1h5M4s).

The Resolution Process

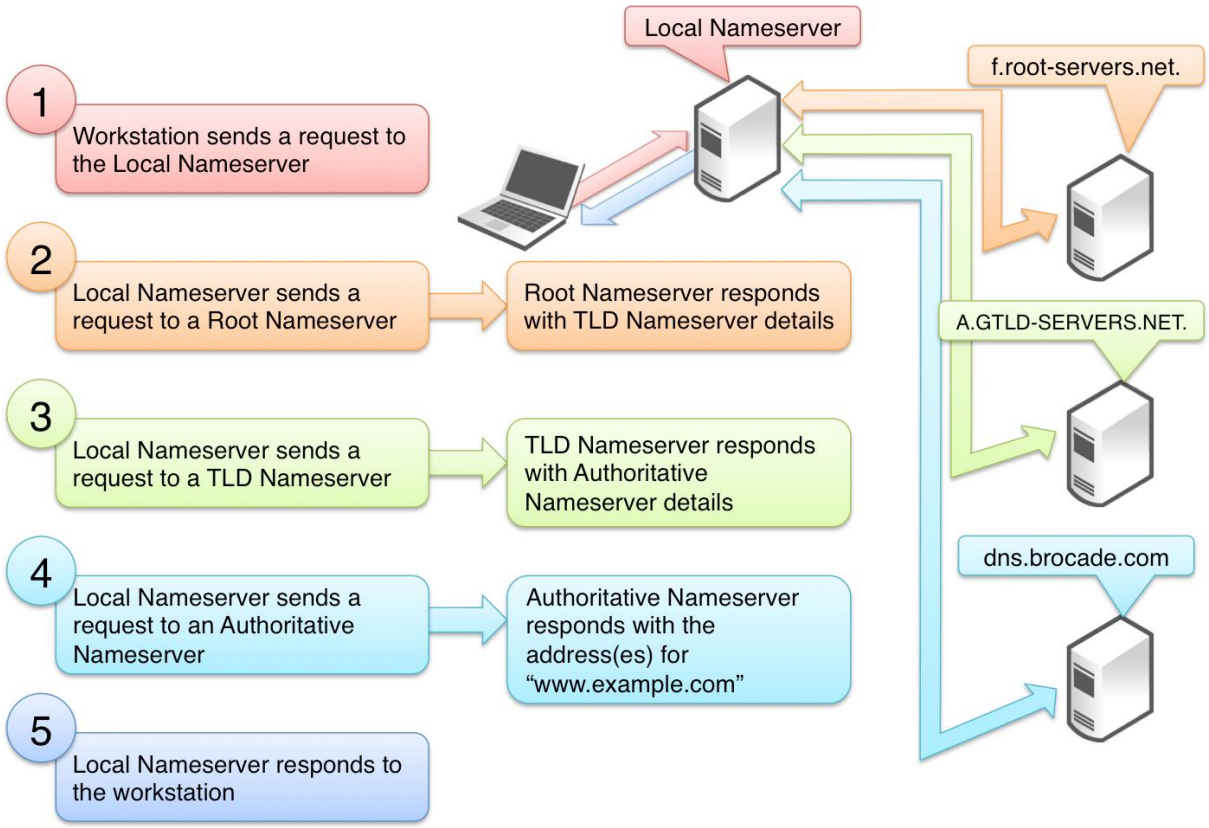

Local Nameserver
f.root-servers.net.
1
Workstation sends a request to the Local Nameserver
2
Local Nameserver sends a request to a Root Nameserver
Root Nameserver responds with TLD Nameserver details
A.GTLD-SERVERS.NET.
3
Local Nameserver sends a request to a TLD Nameserver
TLD Nameserver responds with Authoritative Nameserver details
dns.brocade.com
4
Local Nameserver sends a request to an Authoritative Nameserver
Authoritative Nameserver responds with the address(es) for “www.example.com”
5
Local Nameserver responds to the workstation

Fig 77. The Resolution Process

The following steps detail how a user's client workstation resolves a Web site's URL to a correct IP address:

Note: If you are using the Traffic Manager's built-in DNS server capability, "dns.example.com" in this scenario might correspond to the Traffic Manager as the authoritative DNS server.

---

1. When a user opens a Web browser and types www.example.com into the address bar, their workstation needs to send an HTTP request to a Web Server. Before it does so, it must resolve the domain name of the Web server. Their workstation sends a DNS request to the Local Name Server in order to determine the IP address.

2. The Local Name Server then refers to its Root Hints zone file. The Root Hints zone file contains a list of domain names and IP addresses for the Root Name Servers. The Local Name Server then sends a DNS request for www.example.com to one of the Root Name Servers listed in its Root Hints zone file. "f.root-servers.net." compares the DNS request to the list of zones for which it is authoritative. In this case, it is authoritative for the "." domain. Within the zone file for ".", there are delegation records for all Top Level Domains. The Top Level Domain for www.example.com is ".com.", so "f.root-servers.net." sends a DNS

response back to the Local Name Server containing the domain names and IP addresses for the Name Servers that are authoritative for ".com."

3. A.GTLD-SERVERS.NET[13] is listed (among others) as authoritative for the ".com." domain, so the Local Name Server then sends the same request to it. A.GTLD-SERVERS.NET then responds with a list of name servers authoritative for "example.com.".
4. Since "dns.example.com." is an Authoritative Name Server for the "example.com." domain, the local name server then sends *it* the request. Because "dns.example.com" contains all of the Resource Records for "example.com." in its Zone Files, it provides the IP addresses (in the form of one or more "A" records).
5. Now that the Local Name Server has the IP address, or IP addresses, for www.example.com, it passes on the information to the workstation that made the initial DNS request. The machine that performed the recursive lookup is the Local Name Server.

The type of query the client made is known as "recursive" due to the fact that the local name server searches the DNS tree from top to bottom. A "non recursive" query might also be sent. This means that only a name server authoritative for the domain name being asked about should provide the answer; otherwise, the name server receiving the query should refuse to respond to the query.

In addition to returning the final DNS result to the end user, the Local Name Server usually caches the results of the intermediate lookups. Consequently, not every request requires the full set of DNS requests illustrated above. For this reason, the Local Name Server can be referred to as either a "Recursive Name Server" or a "Caching Name Server".

Supported DNS Features

The Traffic Manager DNS Server is designed to conform to the majority of the DNS technical specifications contained in RFC 1034 and RFC 1035. This section lists the implemented features and exceptions.

Implemented Features from RFC 1034

The Traffic Manager DNS Server implements the following features:

- Authoritative server behavior.
- Domain name space database.

[13] "GTLD" is an acronym for "Global Top Level Domain".

- Positive and negative answers (NODATA, NXDOMAIN, SERVFAIL, FORMAT ERROR, REFUSED, NOT_IMPLEMENTED).
- Delegations (also known as referrals).
- Class IN of the DNS database.
- The ANY record type.
- Wildcard DNS records (see also RFC 4592).

Implemented Features from RFC 1035

The Traffic Manager DNS Server implements the following features:

- Supported record types:
  - A
  - CNAME
  - MX
  - NS
  - PTR
  - SOA
  - TXT
- Other supported record types:
  - AAAA (see also RFC 3596)
  - SRV (see also RFC 2782)
- Supported DNSSEC record types (see also RFC 4034):
  - DNSKEY
  - RRSIG
  - DS
  - NSEC
  - NSEC3
  - NSEC3PARAM
- DNS answers use message compression (see also RFC 1035, section 4.1.4).

- UDP and TCP are fully supported as transport protocols with IPv4 and IPv6.
- Zone text files (also known as "master files"), as defined in RFC 1035 section 5, are supported. The Traffic Manager can process the following aspects of zone files:
  - Definition of resource records of supported types listed in this section
  - The $TTL directive
  - The $ORIGIN directive
  - The use of parentheses and multiline record definitions
  - The @ sign
  - Comments placed after a semicolon (;)
- Reverse lookup using in-addr.arpa domain is supported, as in RFC 1035 section 3.5.

Exceptions for RFC 1034

The following features are not implemented or supported:

- Resource record classes CS, CH, and HS.
- Inverse queries (these queries are made obsolete by RFC 3425).
- Zone transfer.
- Status queries (experimental).
- Completion queries (obsolete).

Exceptions for RFC 1035

The following features are not implemented or supported:

- The $INCLUDE directive.
- The use of the backslash (\) operator to define a literal or control character. For example, "\." to place a dot character in a label.
- \DDD where each D is the octet digit corresponding to the decimal number described by DDD. The Traffic Manager assumes the resulting octet is text and does not check it for special meaning.

Other Implemented Features

The Traffic Manager DNS Server implements the following features:

- Case sensitivity is fully supported.
- RFC 2308, "Negative Caching of DNS Queries", is fully implemented.
- EDNS(0) is implemented, as per RFC 6891, "Extension Mechanisms for DNS". The Traffic Manager makes use of EDNS to handle UDP response sizes that are larger than 512 bytes. The Traffic Manager additionally uses the EDNS "DO" flag to determine whether incoming DNS questions indicate DNSSEC support, as per RFC 3225. No further EDNS features are supported.
- The DNSSEC protocol is supported, as defined in the following RFCs:
  - RFC 4033, "DNS Security Introduction and Requirements"
  - RFC 4034, "Resource Records for the DNS Security Extensions"
  - RFC 4035, "Protocol Modifications for the DNS Security Extensions"
  - RFC 5155, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence"
- For DNSSEC support, note that:
  - The Traffic Manager is able to parse signed DNSSEC zone files.
  - The Traffic Manager is able to answer with DNSSEC signed records in a standards-compliant manner.
  - The Traffic Manager checks the expiry time on DNSSEC signatures in a zone file when the zone is associated with a virtual server, and periodically every hour thereafter. If a signature has expired, or expiry is due to occur within 7 days, the Traffic Manager generates an event through the Alerting system (see CHAPTER 21, "Event Handling and Alerts").

Other Excluded Features

The following features are not implemented or supported:

- Classless delegation, as per RFC 2317, is not supported.
- RFC 3597, "Handling of Unknown DNS Resource Record (RR) Types", is not supported.
- For DNSSEC support, note that:
  - The Traffic Manager does not provide tools to sign zone files.
  - The Traffic Manager does not provide tools to manage the cryptographic keys lifecycle.

Configuring the DNS Server

This section describes how to configure your Traffic Manager as an authoritative DNS server.

Configuration Summary

Use the Admin UI, REST API, SOAP API, or command-line interface to perform the following tasks, each of which is described in detail later:

- Upload your DNS zonefile to the Zonefiles Catalog.
- Create a "zone" configuration to define an "origin domain" and encapsulate the uploaded DNS zonefile.
- Create a DNS-configured Virtual Server and assign to it one or more zones to serve incoming DNS requests.

Your zonefiles can be reused and associated with more than one zone configuration to enable multiple zones to advertise the same hosts under different origin domains. Furthermore, a DNS Virtual Server can utilize any number of zones in order to provide an authoritative DNS service covering many domains.

Note: If you associate a single zonefile with more than one zone configuration, Brocade recommends avoiding redefining the domain origin (using the $ORIGIN declaration) inside the zonefile.

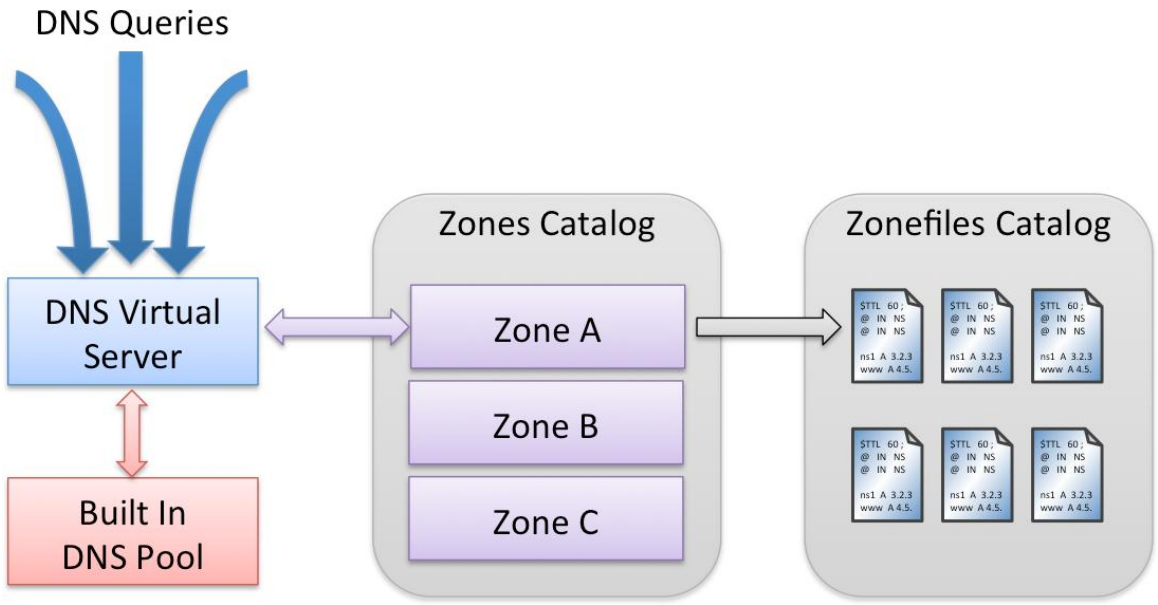


Fig 78. Configuring the Traffic Manager as a DNS server

Using this mechanism, the Traffic Manager itself answers DNS requests using its built-in DNS capability, rather than passing the request on to a pool of external DNS server nodes.

Uploading DNS Zonefiles to the Traffic Manager

A zonefile is a text file containing the resource records that instruct the Traffic Manager what answer to return when it receives a DNS query for a domain. Each zonefile must conform to the format prescribed in RFC 1034 and RFC 1035.

The Traffic Manager stores your zonefiles in the DNS Zonefiles Catalog. Choose Catalogs > DNS Server > Zone Files Catalog to upload a new zonefile.

Note: The Traffic Manager does not validate the contents of your zonefile until you associate it with a zone configuration. Any detected anomalies are added to the event log.

Setting Up Traffic Manager Zones

The Traffic Manager does not directly access and parse uploaded zonefiles. To use a zonefile, first create a *zone* configuration that references a zonefile in the catalog.

Each zone also contains a domain "origin" that the Traffic Manager uses with the resource records in the attached zonefile. In other words, all domains listed in the zonefile branch off of this origin and take the form of <domain>.<origin>.

For example, if your origin is "example.com", you can specify domains "www" and "www.support" in your zonefile to create domain lookups for "www.example.com" and "www.support.example.com".

To create a zone configuration, choose Catalogs > DNS Server > Zones Catalog. Type the name and domain origin for your zone, and select a previously uploaded zonefile to use. Click **Create Zone** to save your zone configuration.

When you create a zone configuration with an attached zonefile, the Traffic Manager reports any syntax issues with the contents of the zonefile in the event log.

Configuring a DNS Virtual Server

You typically create a Virtual Server to handle incoming DNS requests and pass them on to a pool of back-end DNS server nodes. With the internal DNS server capability, you instead configure a DNS Virtual Server to use the built-in DNS pool "builtin_dns", and link to it the desired zone configuration to use for DNS lookups.

Note: To ensure full service coverage, Brocade recommends that you deploy both a DNS (UDP) and a DNS (TCP) virtual server.

To create a DNS Virtual Server:

1. Choose Services > Virtual Servers.
2. In the "Create a new Virtual Server" section, enter the following:
   - **Virtual Server Name**: The identifying name for this virtual server.
   - **Protocol**: Select "DNS (UDP)" or "DNS (TCP)" from the drop-down list.
   - **Port**: Choose the incoming port number that this virtual server listens on (typically "53").
   - **Default Traffic Pool**: Choose "builtin_dns" to instruct the Traffic Manager to use its own DNS capability.
3. Click **Create Virtual Server** to create this virtual server.
4. On the Virtual Server edit page, click **DNS Server** to select one or more zone configurations you want this virtual server to use for DNS lookups.
5. To enable your Virtual Server, set **Enabled** to "Yes" on the Virtual Server edit page or click the *play* icon next to the Virtual Server name on the Traffic Manager home page.

Note: The default traffic pool "builtin_dns" is only available for Virtual Servers that use an internal protocol of DNS (TCP) or DNS (UDP), and is not listed in, or selectable from, your normal pool list.

You can also modify the following settings on the **Virtual Server > Edit > DNS Server** page:

- **dns!edns_udpsize**: The message size advertised in UDP responses under EDNS.
- **dns!max_updsize**: The maximum size allowed for UDP responses.
- **dns!verbose**: To assist debugging efforts, enable this setting to instruct the Traffic Manager to provide a more verbose level of DNS information in the event log. For typical production use, disable this setting.
- **dns!rrset_order**: The DNS record response order. Set to "Fixed" to instruct the Traffic Manager to return DNS records of the same name and type in the order defined within a zone file. Set to "Cyclic" to instruct the Traffic Manager to use a rotating *round-robin* system. That is, for successive responses, DNS records of the same name and type are returned in a cyclical manner - with each record moving one place forward in the response. This method can facilitate more efficient load distribution.

Note: Caching resolvers can, in some cases, preserve the order of DNS records in the answers they receive from authoritative servers.

CHAPTER 30 Global Load Balancing

This chapter discusses the Traffic Manager's Global Load Balancing (GLB) capability.

Note: This functionality might not be available in your Traffic Manager variant. Contact your support provider for details.

Introduction and Prerequisites

GLB is designed to provide business continuity and improved performance across globally distributed locations. This capability is based upon a technique known as Global Server Load Balancing (GSLB). The Traffic Manager uses geographic load balancing and failover functionality to enable the distribution of traffic across these locations based on availability, performance and user-defined rules.

This chapter describes how to plan and configure GLB in your Traffic Manager deployment. To perform the configuration described here, you must be familiar with standard system administration tasks, including networking and DNS administration. For a background on DNS, see "DNS Primer" on page 402.

For further details about the principles of GSLB, see "About Global Server Load Balancing" on page 416. Brocade recommends that you also refer to the background whitepapers and documentation published on www.brocade.com and the Brocade Community Web site at http://community.brocade.com.

About Global Server Load Balancing

Fig 79. Global Server Load Balancing

Global Server Load Balancing (GSLB) is a technique that manages how clients are connected to a particular geographic location when a service is hosted in multiple locations.

- In an "Active-Passive" (or Primary-Backup) configuration, one location is nominated as active for each service. The other locations are idle for that service. If the active location becomes unavailable, one of the passive locations becomes active and all clients are directed to it.
- In an "Active-Active" (or multiple Primary) configuration, all locations are used and clients are load-balanced between them based on location performance and proximity.

The primary purpose of a GSLB system is business continuity, to ensure that services are always available even when one or more service locations becomes unavailable.

A second purpose of GSLB is to improve the customer experience. In other words, the purpose is to load balance each user to the best location from a choice of several. The choice can be based on location performance and proximity, so that clients are directed to the location that is closest and is performing the best. With this capability, the client gets the best possible level of service.

GSLB Within the Traffic Manager

The Traffic Manager implements GSLB techniques as the Global Load Balancer (GLB) feature. The GLB feature causes the Traffic Manager to add location-awareness to DNS lookups performed by end-users, so that the result of the DNS lookup is dependent upon its source.

You can add GLB Services to any Virtual Server using the DNS protocol. When a DNS request is received at this Virtual Server, the Traffic Manager forwards the request to the associated Pool. This might be either the built in internal DNS server pool (see CHAPTER 29, "The Traffic Manager DNS Server") or it might be a pool of external DNS servers. With both of these options, you must configure the pool back ends to return multiple "A" records, containing all IP addresses for the domain names which are to be resolved, across all locations.

If the domain name resolved matches the domain name in a GLB Service attached to the Virtual Server, the Traffic Manager filters the list depending upon the GLB Service configuration, before returning to the end user only the IP address or IP addresses that are appropriate for the end user's location.

You can deploy the Traffic Manager in each GLB Location so it can monitor local performance and availability, although this is not necessary. All monitoring can be performed remotely should you require your Traffic Managers in some alternative location. Your Traffic Managers exchange monitoring information with their peers, regardless of their actual location, so that all Traffic Managers can work together to coordinate their operation.

Note: A GLB Service only filters "A" records, not "AAAA" records. If the DNS response contains any "AAAA" records, the Traffic Manager passes the response through untouched.

Deployment Planning

Traffic Manager Positioning

The Traffic Manager contains a database of geographic locations for the public IPv4 address space and acts either as an authoritative DNS server or as a proxy for a separate back-end DNS service. You deploy your Traffic Managers either in parallel with, in front of, or instead of your current DNS infrastructure. When the Traffic Manager receives a DNS request from a client, it determines the response from either the internal or external DNS service it is managing. The Traffic Manager then modifies the response, based on a number of configurable metrics (including geographic location), before sending it back to the client.

To see more information about the Traffic Manager's built-in DNS capability, see CHAPTER 29, "The Traffic Manager DNS Server".

Deployment Methods

A DNS domain needs a set of name servers that are nominated as authoritative by the name servers for the parent domain. Where the parent domain is a public domain, such as a global top level domain, you modify the authoritative name servers for your domain through a DNS registrar.

To use GLB with a hostname, you can deploy the Traffic Manager in one of two ways:

- With an *Inline* deployment, your Traffic Managers become the authoritative DNS servers for the domain containing the hostname. By modifying the delegation records in the parent domain, you ensure that all DNS traffic for the domain is directed to your Traffic Managers. The Traffic Managers either serve the requests using the built in DNS server capability, or forward it to your existing DNS servers, and then manipulate the responses on hostnames with GLB configured.
- With a *Parallel* deployment, the Traffic Managers are not authoritative for the domain containing the hostname, and you instead create a parallel hostname in a separate DNS domain for which the Traffic Managers are authoritative. In the existing domain, create a CNAME to the parallel hostname in the other domain, and configure a GLB Service with the parallel hostname.

  When your (Authoritative) Name Servers receive a DNS request for the hostname, they respond with a CNAME, which causes the client to make a DNS request for the parallel hostname from a Traffic Manager. As with the Inline deployment, the Traffic Manager either processes the request using the built in DNS server, or forwards it to another authoritative DNS server. The response is then manipulated according to the GLB Service configuration.

A Parallel deployment can be more complex to configure, but is a better solution if you are managing large numbers of domains or if you expect to make frequent changes.

Inline Deployment

For an Inline deployment, the Traffic Manager operates as the authoritative name server for the entire domain containing the hostname. A DNS request is received on a DNS Virtual Server, and processed by the associated Pool, which might be either the built in DNS server or an external pool of DNS servers.

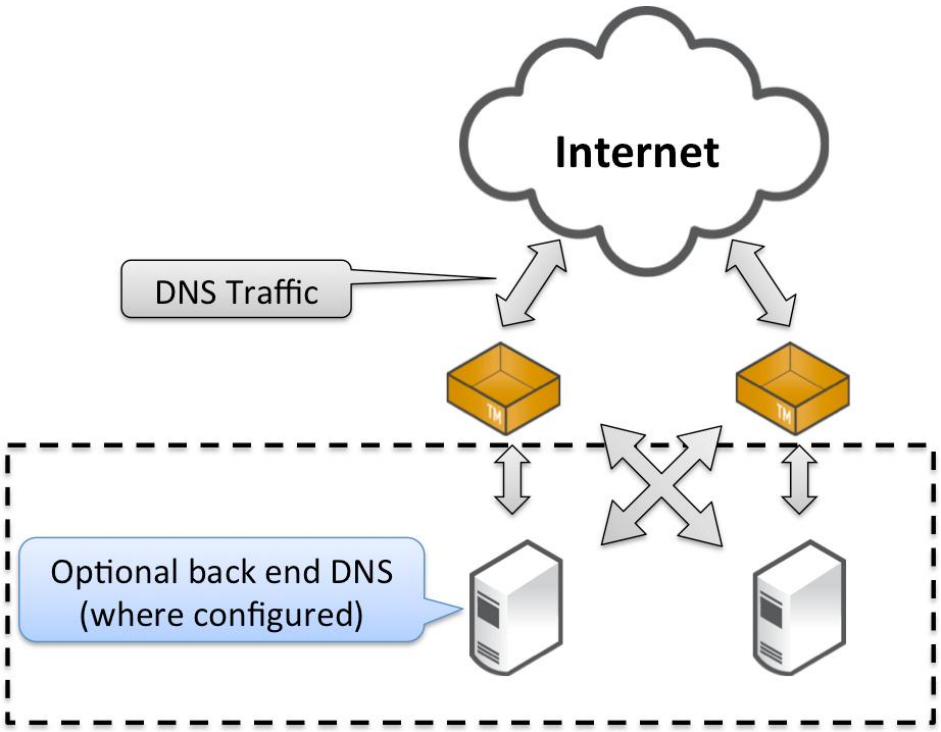


Fig 80. Inline Deployment Diagram

To deploy the Traffic Manager using the Inline method, configure the built in DNS server or external DNS servers to return multiple "A" records for each hostname you want to globally balance. To configure the built in DNS server, see CHAPTER 29, "The Traffic Manager DNS Server".

Then create DNS Virtual Servers on your Traffic Managers. For greatest coverage, Brocade recommends creating both a DNS (TCP) and a DNS (UDP) Virtual Server. Set the default pool to either the built in DNS Server pool or your external DNS servers, depending on your requirements. While using an external DNS pool, the Traffic Managers proxy all DNS requests to these Name servers.

Add a GLB Service to your DNS Virtual Servers for the hostname to be globally balanced, and make the Traffic Managers authoritative for that domain by updating the delegation records in the parent domain.

The parent domain contains a number of record types:

- **NS (delegation) records**: Hostnames of the DNS servers that are authoritative for the domain. You typically provide multiple NS records.
- **A and AAAA (glue) records**: For every NS record, you must provide either an A record or an AAAA record. Particularly if the hostnames in the NS records are contained in your own domain, the parent zone needs to contain glue A or AAAA records, which provide the IP addresses of the hostnames.

To update the parent domain, first determine which organization manages the parent domain. This might be an external DNS registrar, another part of your organization, or a domain that you manage yourself.

To update the delegation records with a DNS registrar

1. DNS settings controlled by your DNS registrar can typically be configured through a Web based interface. Contact your DNS registrar for details about its configuration interface.
2. Login to your domain configuration page and locate the list of name servers.
3. Each record consists of a Fully Qualified Domain Name (FQDN) and, optionally, corresponding IP addresses. To update these records, replace the FQDNs and IP addresses for your current DNS servers with those of your Traffic Managers.

For further assistance with this procedure, contact either your DNS provider or your DNS registrar.

Once your parent domain's delegation records have been updated, the Traffic Manager receives all DNS requests for the hostname to be load balanced. On receiving a request, the Traffic Manager forwards it to the appropriate DNS back end, and receives a list of IP addresses as a response. The Traffic Manager processes this list to select precisely which IP address it returns to the end user.

Parallel Deployment

To create a parallel deployment, you configure a CNAME for your DNS entry that points to a hostname in a separate domain, often a subdomain of the domain that you own. You then configure GLB for the parallel hostname, in accordance with the Inline deployment method.

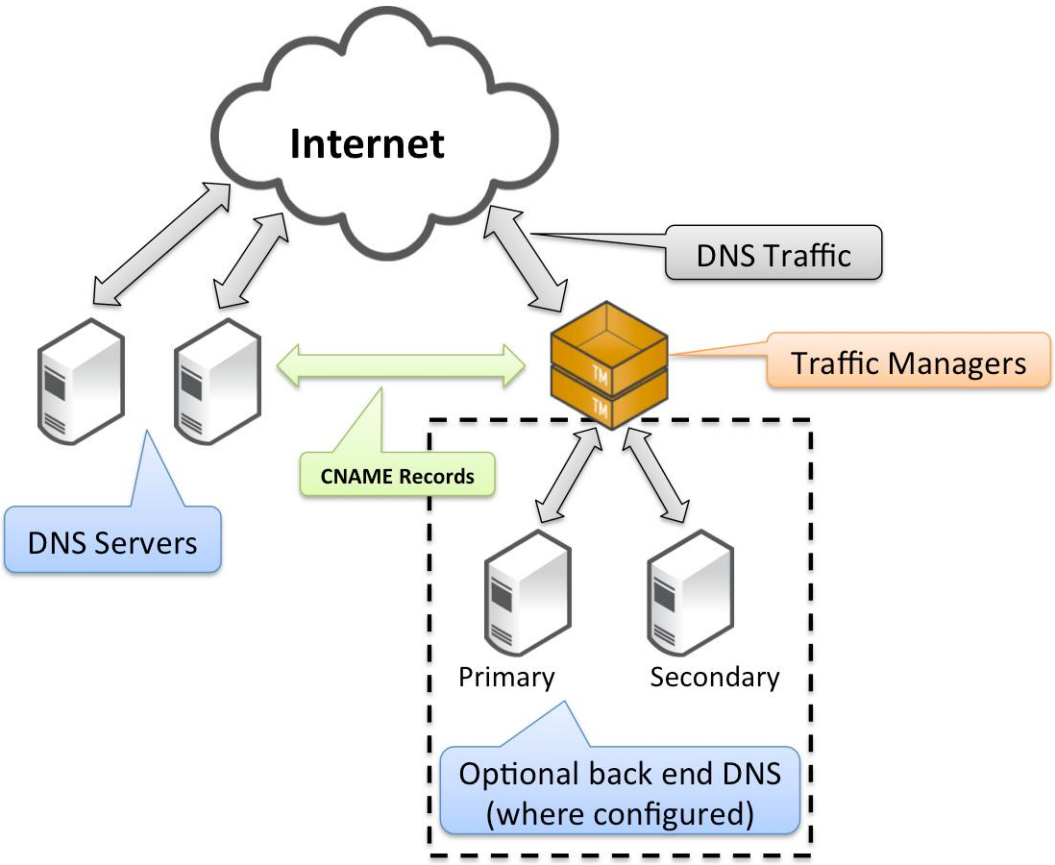


Fig 81. Parallel Deployment Diagram

Suppose, for example, that you manage the domain "example.com" yourself, and want to enable GLB on "www.example.com", using a parallel deployment with the alternative hostname "www.gslb.example.com".

You configure the name servers for "example.com" to fulfill two roles:

- Provide the redirection from "www.example.com" to "www.gslb.example.com".
- Delegate authority for "gslb.example.com" to the Traffic Manager.

Using either the internal Traffic Manager DNS server capability, or through a separate pool of back-end DNS servers, you then host the "gslb.example.com" domain on the Traffic Manager. In the "example.com" zonefile, add the following entries:

```
; Delegation for gslb.example.com using the Traffic Manager.
gslb.example.com.         IN NS  stm1.gslb.example.com.
                          IN NS  stm2.gslb.example.com.

; Glue records for gslb.example.com
stm1.gslb.example.com.    IN A   192.0.2.11
stm2.gslb.example.com.    IN A   192.0.2.12
```

```
; CNAME for www.example.com to parallel GLB-enabled name.
www.example.com.         CNAME   www.gslb.example.com.
```

When a DNS client looks up the domain name www.example.com, it contacts the name server for the example.com domain. This name server returns a CNAME response corresponding to "try www.gslb.example.com instead". Because it is authoritative for the example.com domain, the name server also gives additional information concerning the names and IP addresses of some name servers for gslb.example.com (which, in this case, are your Traffic Managers).

The DNS client now sends a request for www.gslb.example.com to the Traffic Manager, and that Traffic Manager in turn obtains the list of IP addresses for all your GLB Locations (using either its internal DNS service or by querying your back-end DNS servers). The Traffic Manager processes this list, returning back one or more A records containing the IP address or IP addresses of one of the GLB Locations.

Note: If, in a parallel deployment, your DNS servers are not your Traffic Managers, the DNS servers for the parallel hostname (www.gslb.example.com) must not return the unmodified A records for the original hostname (www.example.com). Therefore, Brocade strongly recommends that your top level DNS servers are different name servers from your back-end DNS servers.

It is also possible, although complex, to use split horizon DNS. However, with this approach care is needed to ensure that unmodified responses do not leak out.

If your back-end DNS servers are all Traffic Managers, this problem does not occur. Instead, you must ensure that the GLB Service is configured on all DNS Virtual Servers publicly serving the globally load balanced hostnames.

A parallel deployment can be more complex to deploy because you must create and configure a new DNS subdomain. However, it is a more flexible and suitable technique if you expect to change your DNS configuration frequently, or want to centralize the configuration for many domains as you can use the same subdomain for each one.

For example, you can alias many different domain names (www.example.com, www.mysite.com, and www.example.org) to the same CNAME (www.gslb.example.com), which is in turn hosted by your Traffic Managers. You can then configure many domains centrally from a single CNAME by setting up the back-end DNS service of your Traffic Manager deployment accordingly.

The Time-to-Live (TTL) Field

When you make a change to any of your DNS records, it is possible that the change might not have an immediate effect on your Internet traffic. This is because each DNS record contains a Time-to-Live (TTL) field. The TTL field typically tells downstream DNS servers to cache this record for a given number of seconds, after which time it should send a new request. To ensure that your records propagate across the Internet in a timely fashion, Brocade recommends a TTL value between 30 and 60 seconds.

Components of a Traffic Manager GLB Deployment

This section provides an overview of the Traffic Manager GLB architecture and outlines the key concepts of a working global load balancer.

Brocade recommends that you read this section before configuring your GLB deployment. To configure GLB, see "Configuring GLB" on page 426.

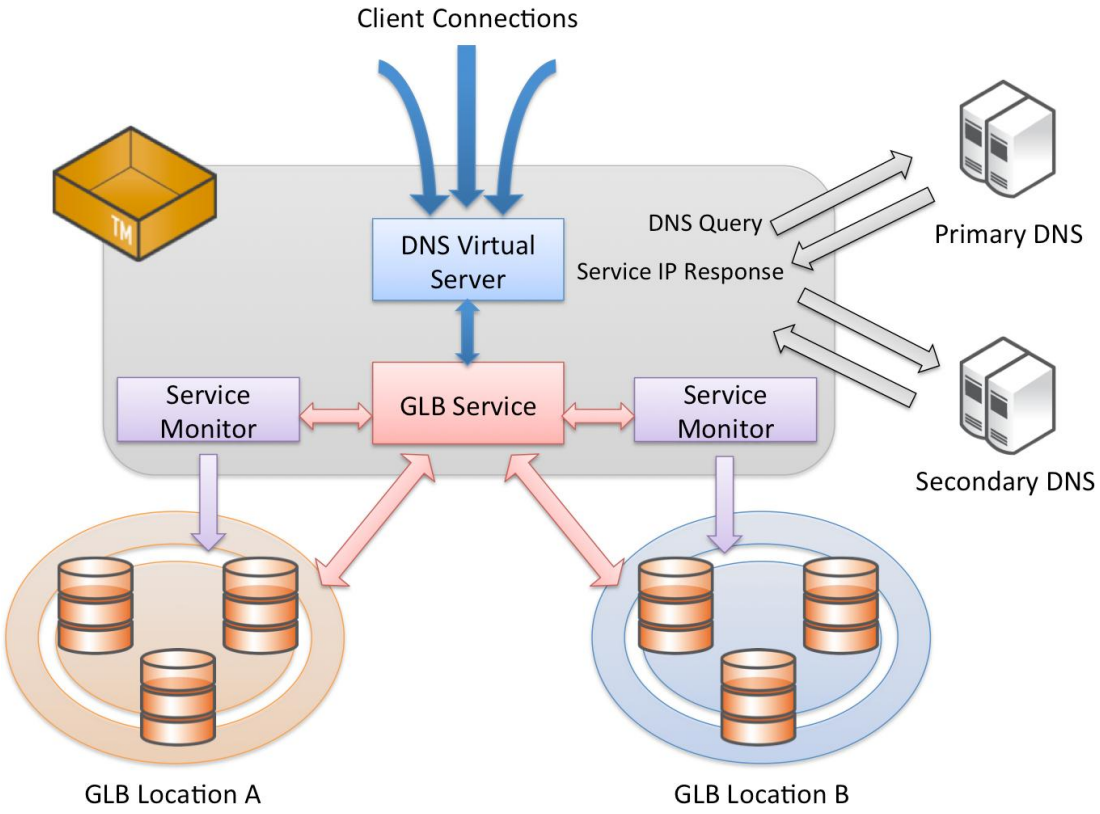


Fig 82. The components of a Traffic Manager GLB deployment

GLB Locations

The Traffic Manager considers GLB Locations as independent geographic service installations, sometimes referred to as datacenters. You might have two or more GLB Locations in different countries around the world, deployed in order to provide a

suitable application delivery infrastructure appropriate to local requirements, or for full disaster recovery for critical systems. Your Traffic Managers do not need to be physically present at a GLB Location, however there are performance and availability benefits for being so.

A Traffic Manager GLB Location is defined by its geographic position in the world. A GLB Location is used by a *GLB Service* to determine where DNS responses should direct clients. Each GLB Location has its own local *DNS Servers* (which the Traffic Manager proxies DNS requests to), and its own *Health Monitors.*

Note: GLB Locations are an independent concept to Configuration Locations, as discussed in "Configuration Locations" on page 394. If you are implementing GLB, a Traffic Manager might be marked as present at a Configuration Location, but this has no bearing on the GLB services it manages.

Traffic managers in different locations can use the same DNS servers. However, for performance and reliability reasons, Brocade recommends that you configure your Traffic Managers to use DNS servers local to them. To achieve this, combine your GLB Locations with the Configuration Location functionality described in CHAPTER 28, "Multi-Site Cluster Management".

GLB Services

A GLB service represents the global load balancing configuration that you want to use for a set of DNS domains. Within a GLB service, you configure the following core information:

- The domains the GLB service applies to.
- The GLB Locations that are hosting the services for those domains.
- The load-balancing algorithm to use for your GLB Locations.

You can also configure additional settings:

- The TTL (Time-To-Live) value, which determines how frequently a DNS client checks the DNS information for the domain.
- Draining: you can drain a GLB Location to stop clients being directed to it.
- Monitors: you can configure one or more Service Monitors to check the correct operation of the services and hosts in each GLB Location.
- Logging: you can log every DNS request and response for inspection and debugging.

GLB Configured Virtual Servers and Pools

Each Traffic Manager listens for incoming DNS requests through a specially configured Virtual Server for your GLB Services. This Virtual Server uses DNS (UDP or TCP) to load balance traffic to a pool consisting of your back-end DNS Server nodes.

If you have multiple GLB Services assigned to a Virtual Server, the Traffic Manager picks the correct GLB Service based on the domain being requested.

DNS Servers

You create a pool containing your DNS Servers as nodes, and the Traffic Manager load balances this pool according to the load balancing algorithm defined within the pool configuration. If one DNS server node times out, an alert is raised and the Traffic Manager handling the request tries the next DNS server.

You might want to install your Traffic Managers in the same physical location as your DNS server(s) for performance reasons, but this is not mandatory.

Service IP Addresses

For the Traffic Manager to manipulate the answers it sends to a client, it must know which IP addresses belong to which GLB Locations. Service IP addresses correspond to the IP addresses that your DNS Servers return, typically using a round-robin mechanism, when queried for the domains you are managing.

Service Monitors

You can configure each individual GLB Service with a list of Service Monitors that should be run in each GLB Location. All of the Service Monitors you define for a GLB Location must succeed for the Traffic Manager to consider that GLB Location in its load balancing decision for that GLB Service.

If any monitor fails (within the failure threshold limits defined in the monitor), the entire GLB Location is marked as *down* and those Traffic Managers that detected the failure avoid this location when load-balancing clients.

Note: In some circumstances, one or more Traffic Managers in your geographically distributed GLB deployment might not detect the same failure as their peers, due perhaps to an isolated network outage. These Traffic Managers can continue to use the GLB Location that their peers have detected as unavailable, provided the Service Monitor returns to them a successful response.

Service Monitors typically test a specific named service IP address or hostname for correct operation. For example, you might configure a Service Monitor to test the primary server load balancer in that GLB Location to verify that it is correctly

serving content. The primary server load balancer has the IP address that is published using DNS for that service in that GLB Location.

You can assign multiple Service Monitors to a GLB Location if you want to test several services or hosts for correct operation.

If you have configured GLB Locations in multiple GLB Services, the Service Monitors for each operate independently. Even though a GLB Location might fail for one GLB Service, the other GLB Services continue to use that GLB Location if their monitors are successful.

Service Monitors can optionally provide performance information to the load balancing algorithm if you are using a "Load" or "Adaptive" load balancing method. The load balancing algorithm then sends more traffic to the GLB Location that reports the best performance.

Performance data is based on the time the Service Monitor takes to complete. In other words, this is the response time of the monitored GLB Location.

Configuring GLB

The Traffic Manager uses the components described in "Components of a Traffic Manager GLB Deployment" on page 423 to perform global load balancing. This section describes how to configure each of these components within a typical Traffic Manager deployment.

Overview

For the Traffic Manager to perform load balancing between geographically separate locations, you must:

- Define the GLB Locations to be served.
- Create Service Monitors to monitor the availability of each of your GLB Locations.
- Create a GLB Service to manage DNS queries for the domains being requested.
- Create a Pool of your back-end DNS Servers.
- Create a DNS Virtual Server to listen for DNS requests.

Defining GLB Locations

To view, modify or add new GLB Locations, use the **Catalogs > Locations** page. To modify the name or geographic position for an existing GLB Location, click the location name or the associated **Edit** button.

Note: Your Configuration Locations and GLB Locations are listed separately on this page, with the ability to add to, and remove from, each list individually. For more information on Configuration Locations, see CHAPTER 28, "Multi-Site Cluster Management".

To create a new GLB Location:

1. Go to **Catalogs > Locations**.
2. In the "Create new GLB Location" section, type an identifying name into **Name** textbox.
3. For **Type**, select "GLB".

   Note: If Multi-Site Cluster Management is disabled, all locations created on this page are of type GLB and this field is not displayed.

4. Click **Add Location** to add this GLB Location and access the location edit page.
5. On the location edit page, choose the **Position** of the GLB Location. Select from either a drop-down list of country names, a specific set of latitude-longitude coordinates, or select a point on the displayed map.
6. Click **Update Location** to save your changes.

To delete a GLB Location, first disassociate it with any GLB Services that are using it. Then click the **Confirm** checkbox and click **Delete Location** to remove it from the system.

Creating a Service Monitor

To configure a Service Monitor for a GLB Location, create a new **pool/GLB** Monitor and instruct it to monitor the IP address or hostname of a service hosted at each of your GLB Locations. The monitor type you choose might require additional parameters - see CHAPTER 14, "Health Monitoring" for more information.

To create a Service Monitor:

1. Navigate to **Catalogs > Monitors**.
2. In the "Create new monitor" section, enter the following:
   - **Name**: The identifying name for this Service Monitor.
   - **Type**: Choose the monitor type that best suits your requirements.
   - **Scope**: Click **Pool/GLB** and enter the IP address or hostname of the service present at the GLB Location you want to monitor. Some monitor types might also require a port number.

3. Click **Create Monitor** to create this Service Monitor and access the edit page.
4. Optionally make any parameter adjustments you need, and click **Update** to save the changes.
5. Repeat this procedure for each GLB Location you want to monitor.

Creating a GLB Service

A GLB Service is based on a set of FQDNs and IP addresses. When you create a new GLB Service, you tell the Traffic Manager which IP addresses are hosted at each GLB Location for each domain name. All of your connected Traffic Managers manage the DNS lookups for each GLB Service.

To create a GLB Service:

1. Navigate to **Catalogs > GLB Services**.
2. In the "Create a new GLB service" section, enter the following:
   - **Service Name**: The identifying name for this GLB Service.
   - **Domains**: Enter at least one domain name you want this GLB Service to balance globally. To enter multiple domains, separate the list with a comma or space.
   - **Add locations**: Click one or more pre-defined GLB Locations you want to host the GLB Service. For information on creating GLB Locations, see "Defining GLB Locations" on page 426.
3. Click **Create GLB Service** to create this GLB Service and access the edit page.

For each GLB Service, use the GLB Service edit page to modify its settings. This page shows the basic GLB Service settings, as entered when you created the service, and all other available configuration settings grouped into subpages.

Each new GLB Service you create is added disabled. To enable your GLB Service, set **Enabled** to "Yes".

To add and remove domains that a GLB Service balances, use the **Domains** table. Create an entry here for every FQDN you want to balance globally.

Use "*" to specify a wildcard character. For example, to manage all DNS lookups for any "example.com" domain, type "*.example.com".

Locations and Monitoring

To configure the GLB Locations and Service Monitors attached to this GLB Service, click **Catalogs > GLB Services > Locations and Monitoring**.

The Traffic Manager presents a dedicated configuration subsection for each attached GLB Location. Click **Update** to save any changes you make.

You typically select each GLB Location you want to use when you first create the GLB Service. To add additional GLB Locations, use the "Add Location" section. To remove a GLB Location, locate and click the dedicated **Remove this location** link in the desired GLB Location configuration subsection.

Note: For a newly created GLB Service, the Traffic Manager might show a warning associated with the GLB Locations you selected. This warning exists until you provide suitable Service IP addresses for each GLB Location.

For each attached GLB Location, you can configure the following:

- **Draining**: To stop sending traffic to this GLB Location, set this to "Yes". For example, if you are physically relocating your equipment from one location to another, or if you are upgrading the backup power system and need to disconnect your UPS, generator, inverter and batteries, you can use this option to instruct the Traffic Manager to stop sending traffic to this GLB Location.
- **Service IPs**: In order for the Traffic Manager to manipulate the answers it sends to a client, it must know which IP addresses belong to which GLB Locations. Use this configuration option to enter a list, or a range, of IPv4 addresses or IPv4 CIDR subnets for this GLB Location. These IP addresses correspond to the IP addresses that your DNS servers return using a round-robin mechanism when queried for the domains you are managing.

  Note: These IP addresses are typically the same IP addresses you configure in the Service Monitors you assign to this GLB Location.

- **Monitors**: Select which Service Monitors the Traffic Manager should use for this GLB Location. Each monitor performs one test, and you can configure multiple monitors for each GLB Location. You can only select monitors that have a host-wide or pool-wide scope. Monitor tests are performed at configured intervals against a specific target, configured in the Service Monitor. When a monitor has exceeded its limit of consecutive failures, the service in a particular data center is flagged as unavailable. Consequently, this GLB Location is filtered out of any subsequent DNS responses.

Load Balancing

To configure load balancing and failure recovery for this GLB Service, click **Catalogs > GLB Services > Load Balancing**.

The available load balancing methods are:

- **Load**: Distributes traffic based on the detected load at each GLB Location. To observe the current load of a GLB Location, the Traffic Manager uses the health monitors you define for each GLB Location. You can override this mechanism and influence load balancing decisions by programmatically setting specific load values for each GLB Location through the Traffic Manager's SOAP API. Use the GLB.Service.setLoad() function to declare a load value for each location. The Traffic Manager instead uses these values when making traffic distribution decisions. To disable the SOAP API and return load reporting to the GLB Location's health monitors, set a load value of 0.
- **Geographic**: Distributes traffic based solely on the geographic location of each client.
- **Adaptive**: Distributes traffic based on both the current GLB Location load and geographic location of the client. This algorithm might be useful if you find that one of your locations is receiving most of the traffic (to the point where it is being overloaded). In this case, this algorithm can help to achieve the best balance between latency due to distance and latency due to load.
  - **Geo Effect**: Use this option to fine tune how much effect the geographic location has when the Traffic Manager decides which GLB Location it should send a client to.
- **Round Robin**: Distributes traffic based on the sequential selection of each available GLB Location in turn. Over time, all GLB Locations receive an equal number of requests.
- **Weighted Random**: Distributes traffic using a random selection generator, weighted by the setting in **Location Weights**. The Traffic Manager sends more traffic to the GLB Location with the greater weight.
  - **Location Weights**: Set the required proportional weighting for each GLB Location. Over time, a GLB Location receives a number of requests equal to the proportion declared using this setting.
- **Primary/Backup**: The Traffic Manager sends DNS records for one primary GLB Location at any time, resorting only to using a backup GLB Location if the primary GLB Location fails. If you provide a service dependent on state information (such as a shopping cart Web site) and you are not replicating your databases globally in real time, this algorithm can be suitable.

  To determine the order of precedence in your list of GLB Locations, use the drag-and-drop tabs to the left of each GLB Location indicator bar. To promote a GLB Location (using a top-down priority order), drag it to the desired place within the stack.

- **Automatic Failback**: Decide what happens when a higher-priority GLB Location recovers after a failure. Click "Yes" to automatically mark the higher-priority GLB Location as active and to instruct the Traffic Manager to direct all users to it. Click "No" to leave the recovered GLB Location as inactive. In this case, the Traffic Manager continues directing users to a currently active GLB Location.
- **Activate best available location**: If "Automatic Failback" is set to "No", click this button to instruct the Traffic Manager to select a new active GLB Location from the list of currently enabled or available GLB Locations.

To set the failure recovery options for all load balancing algorithms, use the following settings:

- **disable_on_failure**: To ensure recovering GLB Locations do not automatically become enabled, click "Yes". Then, to manually enable a particular disabled GLB Location, click **Enable** in the desired GLB Location indicator bar. This setting is useful in deployments where the recovered GLB Location needs to synchronize state or content before it can be used.
- **autorecovery**: If all your GLB Locations fail, click "Yes" to instruct the Traffic Manager to automatically enable the last failed GLB Location when it recovers. If you are using the "Primary/Backup" algorithm, your recovered GLB Location is also made active. Click "No" to leave all recovered locations in a disabled state.

  Note: The "autorecovery" setting, when enabled, takes precedence over "Automatic Failback" and "disable_on_failure".

- **last_resort_response**: If all GLB Locations connected to this GLB Service become unavailable, the Traffic Manager returns the contents of this setting as a last resort DNS response to any client request. Use either a space separated list of IP addresses, a single domain name, or "ALL" to instruct the Traffic Manager to return all IP addresses received from the DNS server.

DNS Authentication (DNSSEC)

To configure the Domain Name System Security Extensions (DNSSEC) settings for this GLB Service, click **Catalogs > GLB Services > DNS Authentication (DNSSEC)**.

The DNSSEC suite of specifications provides a set of security and authentication extensions to DNS. To enable the Traffic Manager to alter DNSSEC authenticated responses, use this page to set up associations between signature domains and DNSSEC Private Keys.

To add an association:

1. Enter a signature domain in the box provided and click the *plus* symbol.

2. Select a private key to be associated with the domain from the drop-down box.

3. Click **Update** to confirm the association.

You can associate multiple keys to one domain, and you can use an individual key to authenticate many domains (a many-to-many relationship).

To add DNSSEC private keys, click **Manage DNSSEC private keys** to access the **Catalogs > SSL > DNSSEC Keys** page. For more information on importing and uploading SSL Certificates and Keys, see CHAPTER 13, "SSL Encryption".

For more information on DNSSEC, see www.dnssec.net.

Rules

To configure TrafficScript rules for this GLB Service, click **Catalogs > GLB Services > Rules**.

You can apply TrafficScript rules to this GLB Service to provide additional traffic management functionality. Note however that you cannot use functions that are incompatible with the DNS protocol (such as http.*). For full details about TrafficScript and its capabilities, see CHAPTER 8, "TrafficScript Rules".

Request Logging

To configure request logging for this GLB Service, click **Catalogs > GLB Services > Request Logging**.

In some circumstances, you might want greater visibility of the DNS requests that are flowing into your Traffic Manager cluster. Request logging lets you log each request into a specified file for later analysis.

You can configure the following:

- **log!enabled:** To enable request logging, set this to "Yes".
- **log!filename:** The full path and filename on the local file system where the Traffic Manager stores request logs for this GLB Service. Click **Macros…** to see the list of available macros.
- **log!format:** The string format of the request log. You can access information about each request using macros escaped by "%". Click **Macros…** to see the list of available macros.

To view the request logs for your GLB Service, click **View Request Logs in file.**

DNS Settings

To configure the DNS settings for this GLB Service, click **Catalogs > GLB Services > DNS Settings**.

You can configure the following:

- **Time To Live (TTL)**: A DNS Server sets a TTL value for each resource record in a DNS response. The TTL value controls how long caching resolvers cache the DNS record before attempting to re-resolve it. GLB services typically require rapid failover in the event of a GLB Location failure and the TTL value set by a DNS server might be too long (for example, several hours or days). To override the TTL for resource records handled by the Traffic Manager, click **Custom** and enter a new value (in seconds).

Creating a DNS Server Pool

For your GLB Virtual Server to perform global load balancing, you need to create a pool containing your DNS Server nodes.

To set up a DNS Server pool:

1. Navigate to **Services > Pools**.
2. In the "Create a new Pool" section, add the following fields:
   - **Pool Name**: The identifying name for the new pool.
   - **Nodes**: A list of one or more DNS Server IP addresses or hostnames, with a corresponding port number in the format <host>:<port>. To specify more than one nodes, use a space or comma separator.
   - **Use autoscaling**: Uncheck this box.
   - **Monitor**: Optionally specify a monitor to check the health of your DNS Servers.
3. Click **Create Pool** to create this pool and access the edit page.

Creating a DNS Virtual Server

Your Virtual Server handles incoming DNS requests and passes them on to your back-end DNS Server pool. You associate the GLB Service you created with this Virtual Server.

To create a DNS Virtual Server:

1. Navigate to **Services > Virtual Servers**.
2. In the "Create a new Virtual Server" section, enter the following:
   - **Virtual Server Name**: The identifying name for this virtual server.
   - **Protocol**: Select "DNS (UDP)" or "DNS (TCP)" from the drop-down list.

- **Port**: Choose the incoming port number that this virtual server listens on.
- **Default Traffic Pool**: Choose the pool containing your DNS Server nodes. For more information, see "Creating a DNS Server Pool" on page 433.

3. Click **Create Virtual Server** to create this virtual server and access the edit page.
4. Click **GLB Services** to access the "Edit GLB Services" page.
5. In the "Add new GLB Service" section, select your required GLB Service from the drop-down list and click Add Service. For more information on creating GLB Services, see "Creating a GLB Service" on page 428.
6. To enable your Virtual Server, set **Enabled** to "Yes" on the Virtual Server edit page or click the *play* icon next to the Virtual Server name on the Traffic Manager home page.

Traffic Visualization

When GLB is enabled, the Traffic Manager provides additional information into the existing visualization and diagnostic tools in the Admin UI. For example, the World Map provides a real time view of incoming DNS requests and the GLB Location to which the client is subsequently connected (for more information, see "The World Map" on page 401).

The Current Activity Graph

To see a real time graph of the activity on your system, click **Activity > Current Activity**.

In the "Settings" section, you can plot your data by Traffic Manager or by Configuration Location. Both choices give you option to select all or some of the corresponding Traffic Managers or locations configured on your system, and whether to plot combined or separate results.

With GLB enabled, the Traffic Manager adds two new sets of metrics pertaining to your GLB Services and Locations to the monitor values tree on the **Current Activity > Edit** page.

For more information on the Current Activity graph, see CHAPTER 7, "Key Features in the Traffic Manager Administration Interface".

The Historical Activity Graph

The Traffic Manager records the system's activity for the past 90 days. To view this data in graphical form, click **Activity > Historical Activity**. You can see the data plotted on either a linear or logarithmic axes.

In the "Settings" section, you can plot your data by Traffic Manager or by Configuration Location. Both choices give you option to select all or some of the corresponding Traffic Managers or locations configured on your system, and whether to plot aggregate or separate results. You can plot a range of timescales from 1 hour to 90 days.

For more details about the Historical Activity Graph, see CHAPTER 7, "Key Features in the Traffic Manager Administration Interface".

The Connections Page

To view information about recent load-balancing decisions the GLB cluster has made, click **Activity > Connections**. For more details about the Connections page, see CHAPTER 7, "Key Features in the Traffic Manager Administration Interface".

GLB Request Logs

To view the individual requests handled by each GLB Service, click **Activity > View Logs** and click on the name of the GLB Service want to observe.

You must enable request logging for a GLB Service before the Traffic Manager can start collecting log information. For more details, see "Request Logging" on page 432.

Testing DNS with DIG

"Dig" is the recommended tool for testing and troubleshooting DNS systems. Dig is maintained by Internet Systems Consortium, Inc., who also maintain the Berkley Internet Name Domain (BIND) software. It is open source software and is available for many architectures and operating systems.

If you are a Linux user, depending on which distribution you run, dig is usually either installed with BIND or provided as supplemental package.

See the Internet Systems Consortium's official Web site for further details:

http://www.isc.org/downloads/bind

An example usage for Dig:

```
$ dig -t any +norec www.example.com @192.0.2.1
```

In the above example:

- "`-t any`": requests any type of record (A, NS, CNAME, PTR, and so on).
- "`+norec`": indicates that the request should be sent with the recursive flag set to "off".
- "`www.example.com`": the record you are testing.
- "`@192.0.2.1`": the name server that the request is sent to.

When testing a DNS server or service, it is very important that you always perform non-recursive queries. This ensures that you receive either an authoritative answer or that you know exactly which name servers are authoritative for the domain name you are trying to resolve. You do not receive an answer that the name server you are querying has looked up for you (since you cannot always verify the source of that information).

Extending the Traffic Manager's GeoIP Database

The Traffic Manager contains a GeoIP database that maps IP addresses to location - longitude and latitude, city, county and country. GLB uses this GeoIP database to estimate distances between remote users and local datacenters.

You can access the database directly using the included `geo.*` TrafficScript and Java functions. For example, to discover the 2-letter country code that a site visitor is accessing from, use the following TrafficScript:

```
$ip = request.getRemoteIP();
$countryCode = geo.getCountryCode( $ip );
```

The Traffic Manager uses a database derived from MaxMind's GeoLite City database (http://dev.maxmind.com/geoip/legacy/geolite/). Brocade updates this database with each Traffic Manager release, however you can update to a later version manually using an upgrade package.

See the Brocade Community Web site for more information at http://community.brocade.com.

Unrecognized IP Addresses

The database does not include locations for private IP address ranges (see RFC1918), and other IP address ranges might be missing or inaccurate if they were recently allocated or moved. This section provides information on how you can extend the internal GeoIP database to add or override IP address ranges.

Extending the Traffic Manager's GeoIP Database

Extensions to the database are stored in the following file `$ZEUSHOME/zxtm/conf/locations.cfg`

Use the following format specified in a single line for each extension:

```
firstIP    lastIP    lat    lon    CC    RR    city
```

The following rules and definitions apply to each mapping:

- The elements in the line are white-space separated.
- The IP address range is inclusive, and the latitude ("lat") and longitude ("lon") are either "-" or decimal degrees.
- CC and RR are either "-" or two-letter country and region codes, for example, US TX for Texas. The special files `ZEUSHOME/zxtm/etc/geo/country_codes.txt` and `ZEUSHOME/zxtm/etc/geo/region_codes.txt` provide a full list of the relevant codes.
- The city name can include spaces, for example "San Francisco", but does not specifically have to refer to a city (any descriptive text is acceptable).
- Only the first two fields are required.

Some example mappings:

```
192.168.0.1 192.168.0.128  52.1234 -0.5678 US TX New datacntr
172.16.0.1  172.16.255.255 -       -       -  -  Test VPN
99.98.97.96 99.98.97.99
```

Testing the IP Address Mappings

You can test any changes with the following example TrafficScript request rule:

```
$text = "";
# Test your own IP addresses with the whereis function here.
$text .= whereis( "192.168.35.40" ); $text .= "\n";
$text .= whereis( "192.168.199.199" ); $text .= "\n";
$text .= whereis( "17.18.19.20" ); $text .= "\n";

http.sendResponse( "200", "text/plain", $text, "" );

sub whereis( $ip ) {
  return $ip . " is in:\n" .
```

```
    "   Country:      " . geo.getCountry( $ip ) . "\n" .
    "   CountryCode:  " . geo.getCountryCode( $ip ) . "\n" .
    "   Region:       " . geo.getRegion( $ip ) . "\n" .
    "   City:         " . geo.getCity( $ip ) . "\n" .
    "   Long/Lat:     " . geo.getLongitude( $ip ) .'/'. geo.ge
tLatitude( $ip ) . "\n";
}
```

Updating Your Traffic Manager Cluster Configuration

You can edit the `locations.cfg` file directly and the local Traffic Manager configuration system notices the fact that this file has changed and automatically accepts the location mappings defined within it.

However, by editing a configuration file directly in the file system, your Traffic Manager configuration is not automatically replicated out to other cluster members. To manually replicate out the updated configuration, use one of the following methods on the Traffic Manager you just updated:

- Use the replicate option on the **Diagnose > Cluster Diagnosis** page in the Admin UI.
- Execute the `ZEUSHOME/zxtm/bin/replicate-config` script on the command line.

CHAPTER 31 FIPS Validation in the Traffic Manager

Introduction to FIPS

This chapter includes information specific to the Federal Information Processing Standard (FIPS) and provides details about how FIPS can be implemented in the Traffic Manager. Contact your support provider if you have any questions.

FIPS is developed by the United States federal government for use in computer systems. If available, and provided you have the required license, the Traffic Manager provides conformity with FIPS through use of a specific operational *mode* and a third party cryptographic operations module.

Important: The Traffic Manager is not *FIPS validated*. Subject to availability, the Traffic Manager uses a validated cryptographic module for all FIPS operations.

FIPS Mode

The primary feature associated with activating FIPS Mode is that it enables the use of a **FIPS 140-2 Cryptographic Module** when processing traffic. In general, the FIPS Mode feature in the Traffic Manager is to help an organization towards FIPS compliance.

Additionally, various configurations/operations of the Traffic Manager are restricted when FIPS Mode is configured, for two reasons:

1. To ensure inputs to the cryptographic module are within the validated range for FIPS 140-2.
2. To ensure that operations that are mutually exclusive with the requirements for FIPS 140-2 cannot be enabled. Examples of restricted operations include SSL 2.0 and SSL 3.0 (only TLS 1.0 and onward are compatible with FIPS 140-2).

FIPS 140-2

FIPS 140-2 is the second revision of the FIPS 140 series publication, published by the National Institute of Standards and Technology (NIST). This is described in detail at the following location:

http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

This document describes the security requirements for cryptographic modules. There are 4 security levels to the FIPS 140-2 requirements, ranging from 1 - 4. It is generally understood that software-only security modules are only capable of

achieving the lowest security level (1); higher security levels have requirements, such as tamper protection, that can only be achieved in a physical product. In general these requirements relate to processes involving cryptographic operations.

In addition, several other NIST "Special Publications" should be considered:

- "Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations" SP 800-52r1 (http://dx.doi.org/10.6028/NIST.SP.800-52r1)
- "Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths" SP 800-131A (http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf):
    - For considerations of what constitutes "key strength" SP 800-51 Part 1 (http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf).

    Note: The Traffic Manager does not implement all recommendations made in this document.

FIPS 140-2 and the Traffic Manager

The FIPS 140-2 cryptographic module used in Linux variants of the Traffic Manager is the Brocade Cryptographic Security Module (certificate number 2099). Details about the FIPS 140-2 validity of this module can be found at the NIST CMVP page, at:

http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2014.htm#2099

When FIPS Mode is enabled, upon start-up of the Traffic Manager the Brocade Cryptographic Security Module will be loaded and initialized. Any failure during initialization (e.g. Power On Self Tests) will be reported as a fatal error - all cryptographic operations in the Traffic Manager *FIPS boundary* will be disabled (for example, SSL decrypting virtual servers and SSL encrypting pools).

The Traffic Manager FIPS boundary

The Traffic Manager uses cryptography in a number of features, the primary example being for SSL/TLS. However, not all of these cryptographic uses are included within the designated Traffic Manager FIPS boundary shown below:

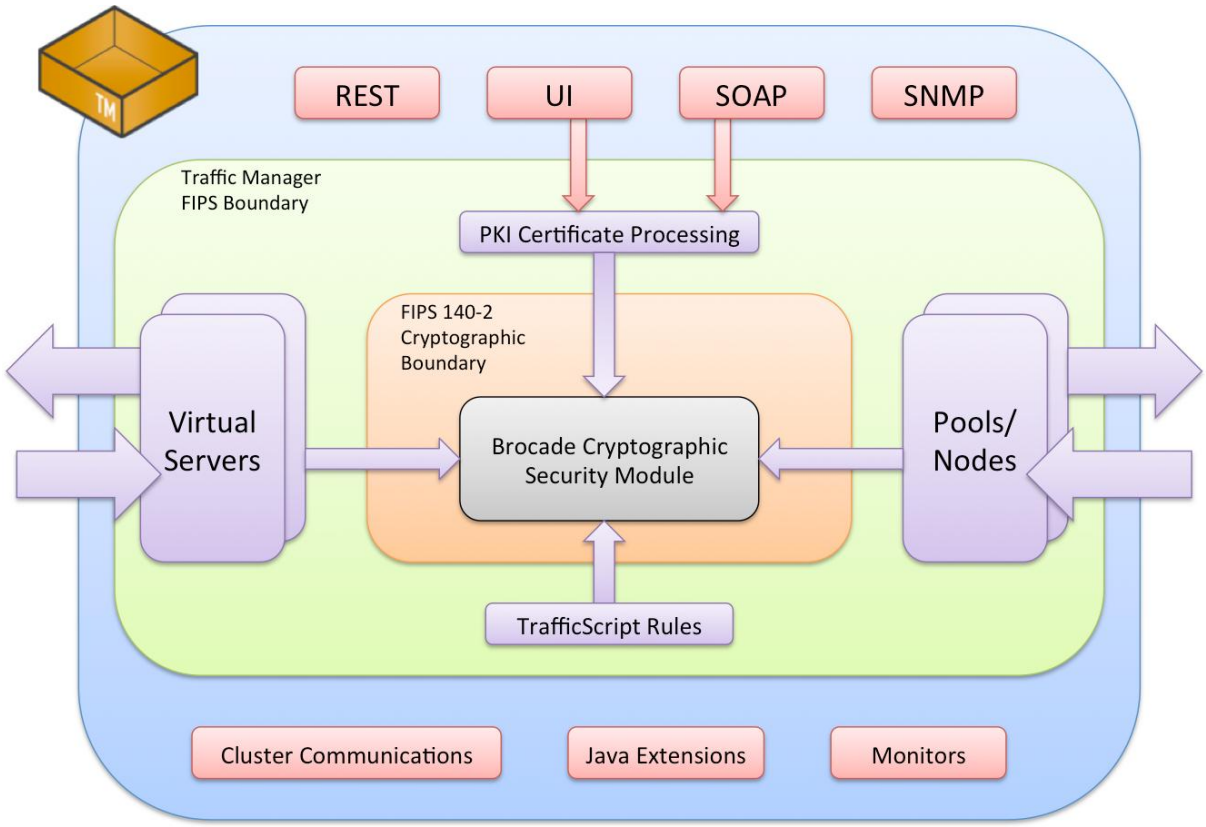


Fig 83. The Traffic Manager FIPS boundary

This boundary is most simply defined as:

1. All the cryptographic operations performed for the processing of traffic.
2. All cryptographic interactions with cryptographic objects used when processing traffic.

When FIPS Mode is in operation:

Item (1) covers primarily TLS; all data that is processed by the Traffic Manager in terms of virtual servers and pools will use the FIPS 140-2 cryptographic module for all cryptography. This includes subordinate operations in TLS such as OCSP requests.

Item (2) considers the manipulation of cryptographic objects stored as part of a Traffic Manager configuration, the primary example being PKI certificates and private keys. Any cryptographic interaction with these objects will be performed with the FIPS 140-2 cryptographic module.

All "control-plane" cryptography is not included within the Traffic Manager FIPS boundary. To clarify, the following, non exhaustive, list of functionality is not within the Traffic Manager FIPS boundary:

- Traffic Manager SNMP interaction;

- The Traffic Manager SOAP API;
- The Traffic Manager REST API;
- The Traffic Manager Admin UI;
- SSH access to a Traffic Manager Appliance.

The importance of (2) in the boundary is that in the case of, for example, the SOAP or Admin UI; although the cryptography used for the transport is not using a FIPS 140-2 cryptographic module, any operations requested that might manipulate cryptographic objects for use within the Traffic Manager FIPS boundary *will be* performed with a FIPS 140-2 cryptographic module.

When considering whether FIPS Mode is suitable for your deployment requirements, you should ensure that you consider whether the Traffic Manager FIPS boundary provides sufficient functional coverage.

TLS as an exception

As mentioned above, the general definition of the Traffic Manager FIPS boundary is that all cryptographic operations performed within that boundary are done using a FIPS 140-2 cryptographic module.

There is an exception to this, where MD5 is required as part of the TLS 1.0 handshake protocol. Without MD5, it would be impossible to inter-operate with TLS clients requiring only the latest version of TLS. This case is covered by a discussion in the "**Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations**" document referenced above, whereby this specific use of MD5 in these very specific parts of the TLS handshake protocol are deemed safe. Refer to this document for further technical details.

As such the Traffic Manager SSL implementation uses an MD5 algorithm from another cryptographic implementation for these specific cases alone, disregarding that MD5 is generally not available from a FIPS 140-2 cryptographic module.

Equally, there are other uses of the MD5 algorithm within the Traffic Manager FIPS boundary, however these have been identified as unrelated to cryptographic or security purposes (i.e. the MD5 algorithm is used as a hash function, rather than as a cryptographic digest).

Deploying FIPS Mode

Preparation

FIPS Mode can only be configured when other certain configuration preconditions have been met. These preconditions are required to provide a suitable environment for using a FIPS 140-2 cryptographic module for processing traffic.

Therefore in preparation for enabling FIPS Mode, the following configuration elements should be inspected and updated accordingly.

Supported SSL/TLS Versions

As per the guidance in SP 800-52r1, only SSL versions TLS 1.0 and later are suitable for operating in an environment using a FIPS 140-2 cryptographic module.

Therefore, to enable FIPS Mode, ensure your Traffic Manager configuration has SSLv2 and SSLv3 disabled by setting "ssl!support_ssl2" and "ssl!support_ssl3" to "No" on the **System > Global Settings > SSL Configuration** page. Additionally, ensure that your virtual servers and pool do not override the global SSL setting on an individual basis (using the "ssl_support_ssl<version>" settings on the **Virtual Server > Edit > SSL Decryption** and **Pool > Edit > SSL Encryption** pages).

Important: If your SSL virtual server clients and pool nodes do not support TLS 1.0 or later, they will no longer be able to connect using SSL.

SSL3 Cipher Suite selection

The FIPS 140-2 requirements provide a list of approved cryptographic methods that a FIPS 140-2 cryptographic module may include. Methods that are not approved are not permitted in the cryptographic module.

Some SSL/TLS cipher suites include cryptographic methods that are not approved by FIPS 140-2, common examples include the RC4 stream cipher and the MD5 digest algorithm. So that all TLS cryptography can be performed using the FIPS 140-2 cryptographic module when in FIPS Mode, the list of cipher suites that can be configured for use with the Traffic Manager is reduced to the following set:

- SSL_RSA_WITH_AES_128_GCM_SHA256
- SSL_RSA_WITH_AES_128_CBC_SHA256
- SSL_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- SSL_DHE_RSA_WITH_AES_128_GCM_SHA256
- SSL_ECDHE_RSA_WITH_AES_128_CBC_SHA256

- SSL_DHE_RSA_WITH_AES_128_CBC_SHA256
- SSL_RSA_WITH_AES_128_CBC_SHA
- SSL_ECDHE_RSA_WITH_AES_128_CBC_SHA
- SSL_DHE_RSA_WITH_AES_128_CBC_SHA
- SSL_RSA_WITH_AES_256_GCM_SHA384
- SSL_RSA_WITH_AES_256_CBC_SHA256
- SSL_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- SSL_DHE_RSA_WITH_AES_256_GCM_SHA384
- SSL_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- SSL_DHE_RSA_WITH_AES_256_CBC_SHA256
- SSL_RSA_WITH_AES_256_CBC_SHA
- SSL_ECDHE_RSA_WITH_AES_256_CBC_SHA
- SSL_DHE_RSA_WITH_AES_256_CBC_SHA
- SSL_RSA_WITH_3DES_EDE_CBC_SHA
- SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA

Important: This will have an impact on interoperability with SSL virtual server clients and pool nodes, if they do not support any of the FIPS 140-2 compatible cipher suites then they will no longer be able to connect via SSL.

TrafficScript Functions

TrafficScript provides access to a number of cryptographic operations for rules executed in the Traffic Manager. When operating in FIPS Mode, these operations will be provided by the FIPS 140-2 cryptographic module.

For TrafficScript functions that use cryptographic methods not approved for FIPS 140-2 cryptographic modules, those functions will not execute successfully when operating in FIPS Mode. It is recommended that you migrate any uses of the TrafficScript functions listed below to an alternative cryptographic digest implementation that is approved for FIPS 140-2 cryptographic modules.

The TrafficScript functions that will produce an error when used in FIPS Mode are those that use the MD5 algorithm:

Function invalid in FIPS Mode	Suggested replacement
`string.hashMD5()`	`string.hashSHA1()`
`ssl.serverCertHash()`	`ssl.serverCertSHA1()`
`ssl.clientCertHash()`	`ssl.clientCertSHA1()`

PKI Certificates

Public Key Infrastructure certificates and keys are cryptographic assets that need to be considered as part of a FIPS 140-2 deployment.

Note: The Traffic Manager only supports PKI using the RSA algorithm.

Key Strength

Although not directly considered in the FIPS 140-2 document, SP 800-131A covers a set of requirements for acceptable strength of keys used in cryptographic operations. These requirements are generally enforced by a FIPS 140-2 cryptographic module, and so must be considered prior to enabling FIPS Mode. For a more formal discussion about what "key strength" actually is (in this context), see the SP 800-51 Part 1 publication referred to in the FIPS 140-2 section above.

Signature Algorithm

Beyond the strength of a key represented by a certificate, a certificate itself contains a cryptographic signature to enable validation that the certificate is the same as that validated by the issuer.

The algorithms used to create and validate the cryptographic signature must be approved for use in a FIPS 140-2 cryptographic module when in FIPS Mode. In FIPS Mode only those certificates whose signature digest uses the FIPS-approved SHA-1, SHA256, SHA384 and SHA512 algorithms can be used.

Traffic Manager SSL Configuration

Prior to enabling FIPS Mode, the Traffic Manager SSL catalog (found at **Catalogs > SSL**) should be audited for:

- Certificates and keys that have a modulus value that is less than 1024 bits, and;
- Certificates that have a signature using a non-approved digest algorithm, typically MD2 and MD5.

If you attempt to enable FIPS Mode in this way, validation errors will highlight any certificates in the current configuration that will prevent the operation from succeeding.

Alternatively, you can use the command-line "cert" tool included in your Traffic Manager installation to evaluate individual certificates. For example, to inspect the SSL Server Certificate 'example' certificate file, use the following command:

```
$ZEUSHOME/admin/bin/cert --format text \
 --in $ZEUSHOME/zxtm/conf/ssl/server_keys/example.public
```

Once these "weak assets" have been identified, you should evaluate their purpose in your Traffic Manager deployment to determine whether they are actually required, and if they are what other systems/services need to be changed/updated/deprecated to successfully move your Traffic Manager deployment towards having FIPS Mode enabled. These "weak assets" must be removed before FIPS Mode can be enabled.

Important: Removing certificates and private keys may have an impact on connectivity with your Traffic Manager deployment. For example, SSL clients may be required to present a certificate signed by a "weak" CA, or the Traffic Manager itself may be required to provide a specific certificate when connecting by SSL to nodes in a pool. In these example cases, removing the "weak assets" will prevent clients, and the Traffic Manager, from creating SSL connections.

Deployment Considerations

In addition to the above, you should consider your deployment where the Traffic Manager is in operation.

When operating in FIPS Mode, the Traffic Manager will reject requests to process cryptographic "weak assets" regardless of whether they are from the local configuration or received from remote systems.

Important: If you are expecting virtual server clients to present certificates when connecting to the Traffic Manager by SSL, or the Traffic Manager is connecting to pool nodes by SSL, you should ensure the certificates presented by virtual server clients and pool nodes are suitable for operations in FIPS Mode.

Certificate Chains

When operating with chains of certificates, *all* the certificates in the chain must not be "weak" in order for that certificate chain to be successfully used by the Traffic Manager in FIPS Mode.

Key Exchange Strength

Some cipher suites make use of the Diffie-Hellman Key Exchange algorithm. SP 800-131A covers a set of requirements for acceptable strength of keys, which is also applicable for the Diffie-Hellman algorithm.

The size of the Diffie-Hellman key used by the Traffic Manager can be configured with the `ssl!ssl3_diffie_hellman_key_length` configuration key (from the *SSL Configuration* section of the **System > Global Settings** page).

Ensuring / Retaining Validity

The validity of a FIPS 140-2 cryptographic module is defined by the testing performed by an accredited testing laboratory. Testing for a software cryptographic module is performed on a specific hardware/operating system platform (operational environment). When a FIPS 140-2 cryptographic module is used in an operational environment where it has been validated, that module can be considered FIPS 140-2 validated.

The details about the operational environments in which the Brocade Cryptographic Security Module (BCSM) has been validated can be found in its listing on the NIST Web site.

While testing has been performed on a limited number of operational environments (and thus the BCSM used in those environments have explicit FIPS 140-2 validity), it is understood that a validated FIPS 140-2 cryptographic module (Level 1) may retain its validity when operating in an environment where it has not been explicitly validated (for FIPS 140-2).

The Implementation Guidance for FIPS 140-2 (http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf) covers this case under section G.5. This covers the "porting" rules that a vendor (Brocade in the case of the BCSM) or a user, for example, an administrator for a Traffic Manager installation, should conform to in order to retain FIPS 140-2 validity on a platform that has not been explicitly listed as validated. Note that NIST (specifically CVMP) remains the overall arbitrator of whether a FIPS 140-2 cryptographic module is and is not considered valid.

Important: If you are concerned about whether or not the Traffic Manager will retain the FIPS 140-2 validity of the BCSM in your installation, please contact your support provider.

Non-Linux Operating Systems

The BCSM has only been tested for validation on versions of the Linux Operating System. Support and validation for non-Linux systems, such as Solaris/SunOS, is not available.

Enabling FIPS Mode

Providing you have followed the preceding instructions for preparing your Traffic Manager deployment, you can enable FIPS Mode by accessing the **System > Global**

Settings page of the Admin UI and setting the `fips!enabled` configuration key to **Yes**.

Note: `fips!enabled` is a cluster-wide setting that cannot vary by location.

Restart Traffic Managers

Setting the `fips!enabled` key in your configuration is not sufficient to have the Traffic Manager operate fully in FIPS Mode; a restart of each Traffic Manager instance in the cluster is required.

Important: Restarting the Traffic Manager disconnects any active connections. For high availability deployments, Brocade recommends that you schedule this operation to minimize the interruption to your services. For further information, see the "Installation and Getting Started Guide" applicable to your product variant.

Validation

When using the Admin UI or SOAP-based Control API after enabling FIPS Mode, the Traffic Manager will perform a validation of your configuration to ensure various criteria for FIPS Mode.

Any aspect of the configuration that is in conflict with FIPS Mode will generate an error. If you see such an error, please refer to the Preparation section above with respect to the error you observe.

Operating in FIPS Mode

When operating in FIPS Mode, each Traffic Manager in a cluster will report whether it is successfully operating in FIPS Mode in the event log. A message such as the following can be seen:

```
FIPS 140-2 Cryptographic module loaded: FIPS Mode in
operation
```

FIPS Mode Errors

If there is an error when trying to enable FIPS Mode, the Traffic Manager will report this in the event log. In the Traffic Manager component where FIPS Mode failed, cryptographic operations (and services that utilize those operations) will be disabled within that component.

As an example, consider a Traffic Manager with two HTTP virtual servers, one with SSL decryption and one without. If FIPS Mode fails to successfully initialize, the SSL decrypting virtual server will be disabled and no traffic will be processed, the other virtual server will operate normally.

Important: Even if a Traffic Manager is successfully started with FIPS Mode in operation, it is possible, although rare, that during operation the requirements for FIPS 140-2 will cease to be met. In these cases, the same behavior as encountered during an initialization failure can be expected: the failing Traffic Manager will report an error, and cryptographic operations will no longer be available where FIPS Mode has failed.

CHAPTER 32 Kerberos Constrained Delegation Support

The Kerberos Protocol

The Kerberos protocol allows entities (known as *principals*) to authenticate with each other through the commonly understood and trusted Key Distribution Center (KDC) service. The KDC for a realm authenticates principals associated with that realm by verifying credentials provided by the principal. It can subsequently generate tickets that the principals operating in a realm can exchange in order to establish authenticated peer-to-peer exchanges of information. A ticket issued for a user to a principal by the KDC contains identity data for the user encrypted using a secret shared only by the principal and the KDC.

Note: Microsoft implementations of the Kerberos protocol refer to the realm as a *domain*, and to the KDC service as the *domain controller*.

Kerberos Protocol Transition and Constrained Delegation

Where clients have direct access to the KDC for the realm their principal is a part of, Kerberos authentication between "Kerberized" applications operates normally. However, where direct access to the KDC is not possible, a user cannot always access services that require the exchange of Kerberos tickets for authentication. For example, a laptop computer solely connected to the public Internet might not have direct access to the laptop user's KDC due to firewall rules. To address this, two extensions to the Kerberos protocol are available: Protocol Transition and Constrained Delegation.

Protocol Transition

This mechanism allows a suitably privileged Kerberos service to obtain a ticket to itself for an arbitrary user principal in a given realm. The KDC expects the service to perform authentication through some other means to confirm the identity of a user before then establishing a ticket for the user in the Kerberos protocol. In other words, the service provides a transition from one authentication protocol to Kerberos.

Constrained Delegation

Since the service providing the protocol transition should have already satisfied itself that the user is who they claim to be, having a ticket for an arbitrary user principal to

itself is of limited use. Through constrained delegation, the service can delegate the ticket for itself to another service, meaning that this ticket can be forwarded on by the service (with a request associated with the original user) to a further Kerberized service. This new service then processes the request in the context of the original user.

Arbitrarily being able to impersonate any user in a realm could be considered too much power for a service, but the "constrained" aspect of the feature allows the KDC to impose a restriction on which service the ticket might be delegated to.

Rationale for Using Kerberos

In the context of the Traffic Manager, Kerberos Protocol Transition can be useful where the Traffic Manager sits on the border of your internal network, providing access to internal services. The internal services on a corporate LAN that authenticate using Kerberos can be made available over a more public network where access to the KDC is not permitted. The Traffic Manager uses alternate means to identify the principal name of the user and perform protocol transition.

The Traffic Manager supports protocol transition from TLS client certificate-based authentication.

Configuring Kerberos on the Traffic Manager

The Traffic Manager's Kerberos support allows it to authenticate a user through a TLS client certificate. The Traffic Manager communicates with a realm's KDC using its own service principal and acquires a ticket to a service running on a back-end node on behalf of a client making a request on a virtual server.

Traffic Manager Service Principal

To participate in the Kerberos realm, the Traffic Manager must have a suitably privileged principal with which to identify itself. You must ensure your identity management system and associated KDC configuration has a Kerberos principal for the Traffic Manager, and is configured to allow protocol transition and constrained delegation to the services the Traffic Manager is managing.

For each Traffic Manager within the cluster, the KDC administrator must configure the KDC to allow the Traffic Manager’s service principal to delegate to other services, and the service principal name must match the name used by the Traffic Manager. This is a string in the form:

<service>/<Traffic Manager hostname>@<REALM>

In this string, use the <service> and <REALM> values from the Kerberos Principal record you want to use from your Kerberos Principals catalog.

For example, if the hostname of the Traffic Manager is "stm1.example.com", and a connection uses a Kerberos Principal record with service name "HTTP" and realm "EXAMPLE.COM", the KDC must trust the principal named "HTTP/stm1.example.com@EXAMPLE.COM" for delegation to your end services.

For Microsoft implementations, this means that you have created a computer account for the Traffic Managers on the domain, have configured a server principal name for the relevant services on the Traffic Managers, and have permitted delegation for those services to the Traffic Managers.

For a Traffic Manager to authenticate itself as the created principal, you must specify the appropriate credentials. For user principals, you specify your credentials in the form of password entry. Since interactive password entry is not convenient for service principals, you must instead use a "Keytab" file for the credentials of your Traffic Manager Service Principals. A Keytab file can contain credentials for one or more principals.

To configure the Traffic Manager to perform protocol transition in a Kerberos realm, upload a Keytab file containing the credentials for all Traffic Managers in the cluster, and configure a Kerberos Principal to use the Keytab file.

Your Keytab files are stored in the Traffic Manager Keytab catalog. To upload Keytab files to the catalog, use the **Catalog > Kerberos > Kerberos Keytabs** page in the Admin UI.

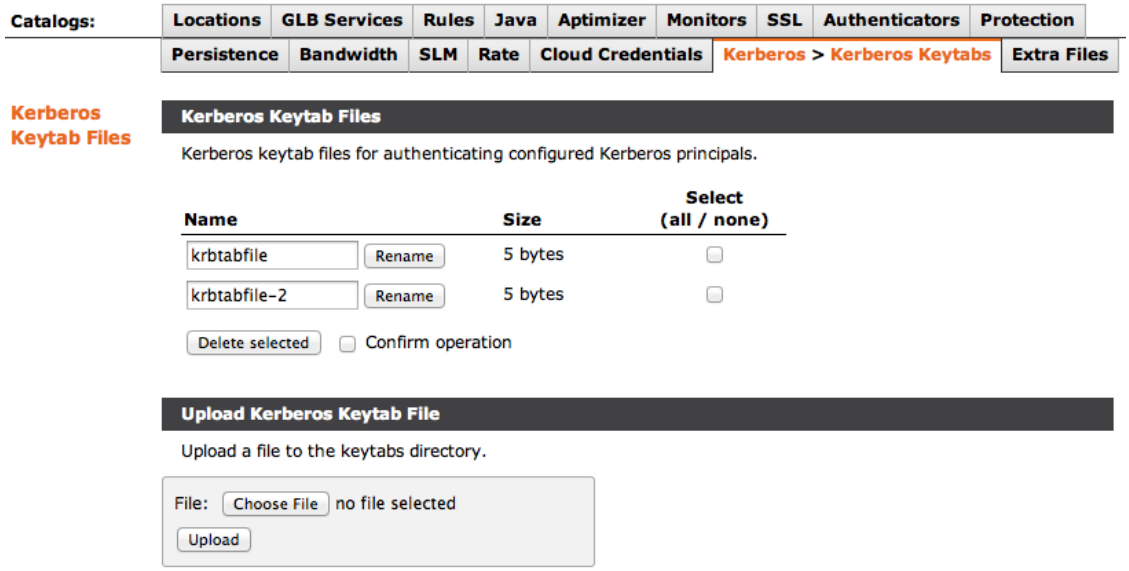


Fig 84. The Kerberos Keytabs Catalog

To create a Kerberos principal catalog item, use the **Catalog > Kerberos > Kerberos Principals** page of the Admin UI. For each principal, specify the following fields:

Configuration Item	Description
name	The identifying name of the Principal configuration.
service	The service name aspect of the Kerberos Principal name that the Traffic Manager uses to authenticate itself.
keytab	The name of the Keytab file in the Kerberos Keytabs catalog that contains the credentials for this principal.
realm	The realm the Kerberos principal is a part of.
kdcs	An optional list of <host>:<port> pairs to explicit specify the KDC services to use for the realm. If you do not specify a list, the Traffic Manager uses DNS to determine the KDCs that should instead be used.
krb5conf	The name of a predefined Kerberos configuration file to use with this Principal (optional). To use a Kerberos configuration file, you must first upload it using the **Catalog > Kerberos > Kerberos Configuration Files** page of the Admin UI.

Virtual Server Protocol Transition Configuration

To use Kerberos Protocol Transition with your virtual server, use the **Virtual Servers > Edit > Kerberos Protocol Transition** page.

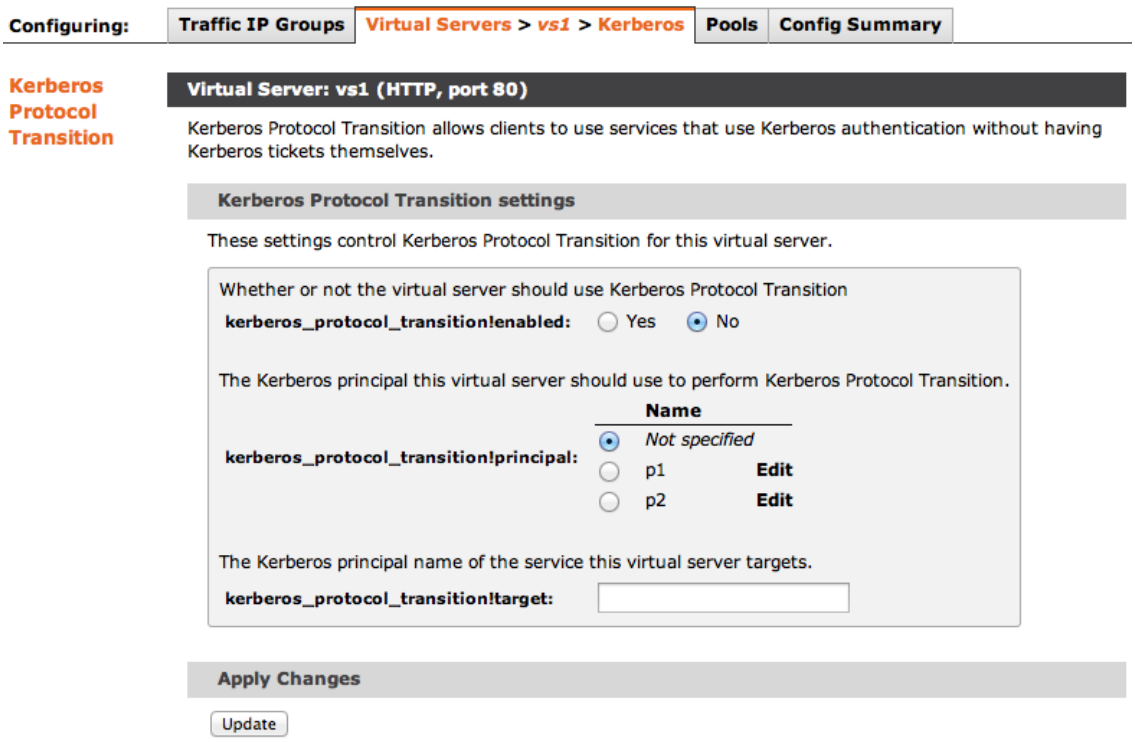

Configuring:
Traffic IP Groups
Virtual Servers > vs1 > Kerberos
Pools
Config Summary
Kerberos Protocol Transition
Virtual Server: vs1 (HTTP, port 80)
Kerberos Protocol Transition allows clients to use services that use Kerberos authentication without having Kerberos tickets themselves.
Kerberos Protocol Transition settings
These settings control Kerberos Protocol Transition for this virtual server.
Whether or not the virtual server should use Kerberos Protocol Transition
kerberos_protocol_transition!enabled:
Yes
No
The Kerberos principal this virtual server should use to perform Kerberos Protocol Transition.
Name
kerberos_protocol_transition!principal:
Not specified
p1
Edit
p2
Edit
The Kerberos principal name of the service this virtual server targets.
kerberos_protocol_transition!target:
Apply Changes
Update

Fig 85. The Virtual Server Kerberos Protocol Transition page

To enable Kerberos, set **kerberos_protocol_transition!enabled** to Yes. Using this page, you can optionally configure a Kerberos Principal and the principal name for a target service the virtual server is handling.

Note: For protocol transition to be successful, you ***must*** configure both a Traffic Manager and a target principal. Configure these settings in either a virtual server or a pool.

The Traffic Manager must acquire the client principal from a client SSL certificate. To enable this, configure your virtual server with **ssl_decrypt** enabled along with **require_cert** to ensure a client certificate is solicited from the client. To ensure the client is properly authenticated, configure your virtual server with appropriate certificate authorities and OCSP settings.

Pool Protocol Transition Configuration

You can also configure Kerberos Protocol Transition using the **Pool > Edit > Kerberos Protocol Transition** page.

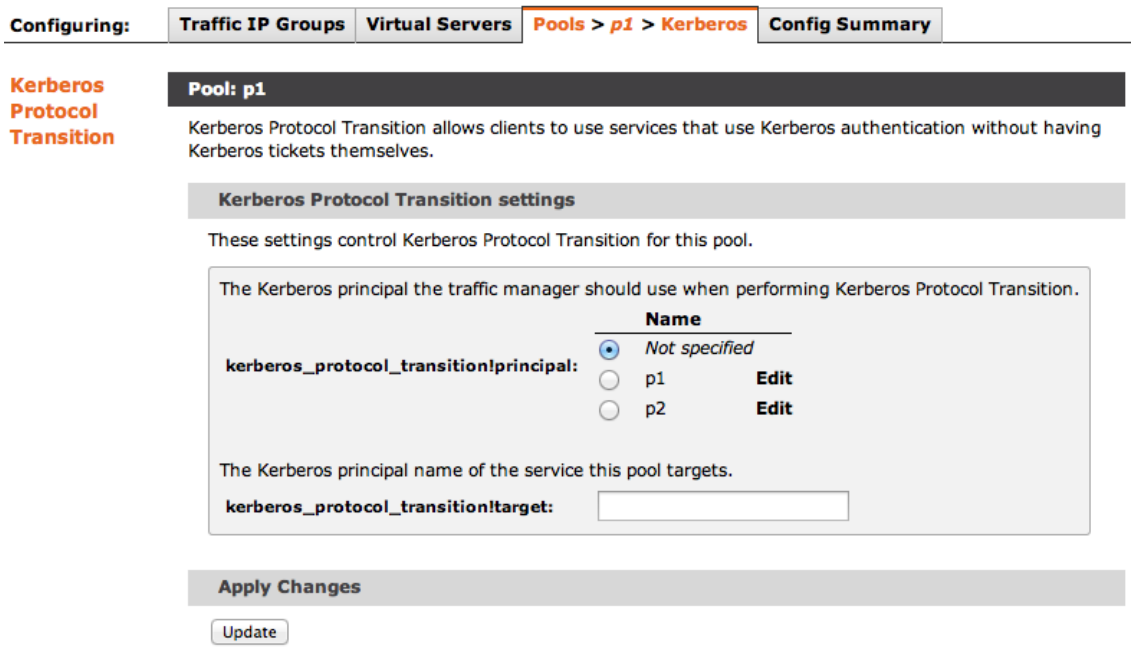


Fig 86. The Pool Kerberos Protocol Transition page

If you specify a principal or target here, this configuration overrides any Kerberos configuration present in a virtual server that uses this pool.

CHAPTER 33 Troubleshooting

This chapter describes how to test that your Traffic Manager installation is working properly, and details the ways to deal with any problems you might encounter.

Tools and Techniques

If you are testing your Traffic Manager configuration, there are a range of tools and techniques you can make use of to understand your configuration, and the behavior of your network traffic:

- **Diagnosis:** The **Cluster Diagnosis** page runs a number of tests across your Traffic Manager system, checking network connectivity, health monitors, configuration errors and conflicting configuration.
- **Event Log:** The **Event Log** (displayed on the Home Page and the Event Log Page) records any significant events that have occurred.
- **Connections:** The **Connections** report gives an instant display of ongoing and recent connections: where they came from, how they were handled, how much data was transferred, etc.
- **Request Logging:** The Traffic Manager can write full logs so that you can record all the details about each request and response for later analysis.
- **Configuration Summary:** If you have a sophisticated configuration, the configuration summary describes what configuration objects are in use, and how they are related.
- **Audit Log:** The **Audit Log** records all configuration changes; when they were made, which user made them, and where the user came from.

This chapter describes each of these tools and techniques in detail.

Diagnosis and Event Logging

If you encounter any problems with your Traffic Manager system or configuration, the first step should be to check the **Cluster Diagnosis** page in the Traffic Manager Admin Server.

Important: If you cannot access the Admin UI on any of the Traffic Managers in your cluster, Brocade recommends starting or restarting the Traffic Manager software. This process is described in the "Installation and Getting Started Guide" applicable to your product variant. Take careful note of any error messages, particularly any messages appended to the error log file in `ZEUSHOME/zxtm/log/errors`.

Click the **Diagnose** button on any page of the Admin Server interface to view the current system status. The **Diagnose** page is divided into sections that help you understand and resolve any issues you might have.

The **Cluster Diagnosis** section reports any problems the Traffic Manager finds with your setup, such as unavailable machines or bad configuration.

The **Event Log** shows the contents of the error logs on each Traffic Manager machine. You can view logs created in a recent period of your choice. The Event Log page dynamically updates as new event messages are added to the log.

The **Audit Log** shows recent configuration changes; when they were made, and which user made them.

The **Technical Support** section provides useful links to various resources, and an option to download technical support data for the Brocade Support team if necessary.

Note: If you are using a Traffic Manager instance within a **Brocade Services Director** deployment, use the *Support Entitlement* identifier displayed in this section when requesting assistance. Brocade uses this information to uniquely identify your level of support entitlement.

If a problem is detected during normal usage, the **Status Applet** will indicate a warning or error condition. Visit the **Cluster Diagnosis** page for full details about the problem.

Fig 87. 'No errors' and 'Errors found' in the cluster warning screens

Monitoring Requests and Responses

If the **Cluster Diagnosis** page does not report any errors that are related to the problems you have observed, the next step is to verify that the Traffic Manager is correctly managing your traffic – reading requests on behalf of the back-end servers, and returning their responses back to the remote client.

You may find the suggestions in the *Generating Test Requests* section of CHAPTER 28 useful when attempting to reproduce any problems.

Connection Activity Report

Click the **Activity** button in the Traffic Manager Admin Server and select the **Connections** tab. The **Connections** page gives a report of all ongoing and recently completed connections that the Traffic Manager has managed. Unfolding the **Key** section gives you details about the information shown.

You can see which Traffic Manager is currently handling the connection and the back-end node involved; the current state of the connection; and the virtual server, rule and pool currently involved with the connection. You can also see the retry count, idle times and byte counts for the connection.

To keep the data up to date, click the refresh button in your Internet browser. You can also download the data as a `.tsv` file.

Using the Connections Report

Note: Verify that the Traffic Manager has managed the requests you are testing. If you do not see your test requests in this list, it is probable that the Traffic Manager never received the requests. Check your client configuration and DNS settings to verify that your test client is sending the requests to the Traffic Manager IP address, and that the request is then managed by one of your back-end servers.

One common error is that the back-end servers issue self-referential responses. For example, suppose that the Traffic Manager is managing traffic to `www.example.com`, and the back-end servers are named `server1.example.com` and `server2.example.com`. One of the servers issues a response containing a link or redirect to `server1.example.com`. The client would subsequently try to contact the back-end server directly, bypassing the Traffic Manager and causing problems. Other protocols which embed DNS names or IP addresses in the response are also prone to this type of problem.

You can increase the size of the Connections report using the **System>Global Settings** page; look in the **Logging** section for the **recent_conns** key.

Request Logs

You can use the Request Logging facility of a virtual server to log many aspects of a transaction. This is very useful for long-term testing, if errors are intermittent or hard to replicate. Please refer to the *Request Logging* section of CHAPTER 4 for configuration instructions.

if you have complicated TrafficScript logic, you can store debugging information for the connection, and log it in the request log. Use the `connection.data.set()` function to store the specific data for the connection in a TrafficScript rule, and then log that data in the request log with the '%{Key-name}d' macro.

Advanced Logging

You can use a TrafficScript rule to log information about the transactions managed by your virtual server.

TrafficScript functions can retrieve information about every part of a request. Functions such as `http.getHeader()` and `http.cookie()` give information about HTTP requests; functions such as `ssl.getSessionID()` provide details about SSL traffic. For other protocols, you can use the `connection.getData()` function to return a specified number of bytes of a request.

You can use the `log.info()` function to write details of the incoming requests to the Traffic Manager's event log file, to pinpoint any problems that are occurring with incoming traffic, or you can use the `event.emit()` function and configure an appropriate Event Handler to give you more control over where debugging messages are recorded.

See the **TrafficScript Reference** for details about the TrafficScript functions.

Important: You should take care when logging such information: if you do this for every incoming request, the log file will grow very rapidly.

Monitoring Events

The TrafficScript `counter.increment()` function is used to count how many times a specific event occurred. For more information, please refer to the *Activity > Current Activity* section of CHAPTER 7.

Detailed Debugging of Connections

If a connection fails, the information logged in the connection activity report and request log will be incomplete because the connection did not complete normally. Connections may fail because of a variety of reasons, including protocol errors, timeouts or unexpected TCP closes.

To debug such unexpected problems, enable the **log!client_connection_failures** and **log!server_connection_failures** settings in the **Virtual Server > Connection Management** configuration page for the affected service, as described in the *Handling Errors* section of CHAPTER 4. These settings will configure your Traffic

Manager to write detailed debug messages to the Event Log whenever a connection fails:

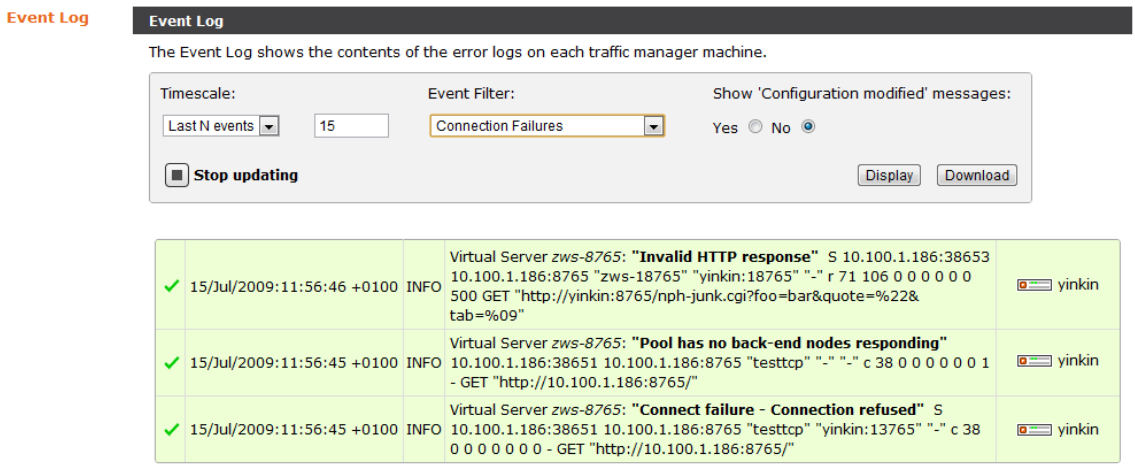

Event Log
Event Log
The Event Log shows the contents of the error logs on each traffic manager machine.
Timescale:
Last N events
15
Event Filter:
Connection Failures
Show 'Configuration modified' messages:
Yes No
Stop updating
Display
Download
15/Jul/2009:11:56:46 +0100 INFO Virtual Server zws-8765: "Invalid HTTP response" S 10.100.1.186:38653 10.100.1.186:8765 "zws-18765" "yinkin:18765" "-" r 71 106 0 0 0 0 0 0 500 GET "http://yinkin:8765/nph-junk.cgi?foo=bar&quote=%22&tab=%09" yinkin
15/Jul/2009:11:56:45 +0100 INFO Virtual Server zws-8765: "Pool has no back-end nodes responding" 10.100.1.186:38651 10.100.1.186:8765 "testtcp" "-" "-" c 38 0 0 0 0 0 0 1 - GET "http://10.100.1.186:8765/" yinkin
15/Jul/2009:11:56:45 +0100 INFO Virtual Server zws-8765: "Connect failure - Connection refused" S 10.100.1.186:38651 10.100.1.186:8765 "testtcp" "yinkin:13765" "-" c 38 0 0 0 0 0 0 0 - GET "http://10.100.1.186:8765/" yinkin

Fig 88. Connection failure reports in the event log

You can filter the event log to only display these messages using the 'Connection Failures' Event Filter as illustrated above.

The log line contains a description of the error (for example, "Invalid HTTP Response") and details that describe the nature of the connection (node used, connection times, retries, etc). The Online Help for the **Virtual Server > Connection Management** configuration page describes the log format in detail.

Click the description to unfold a more readable representation of the connection error:

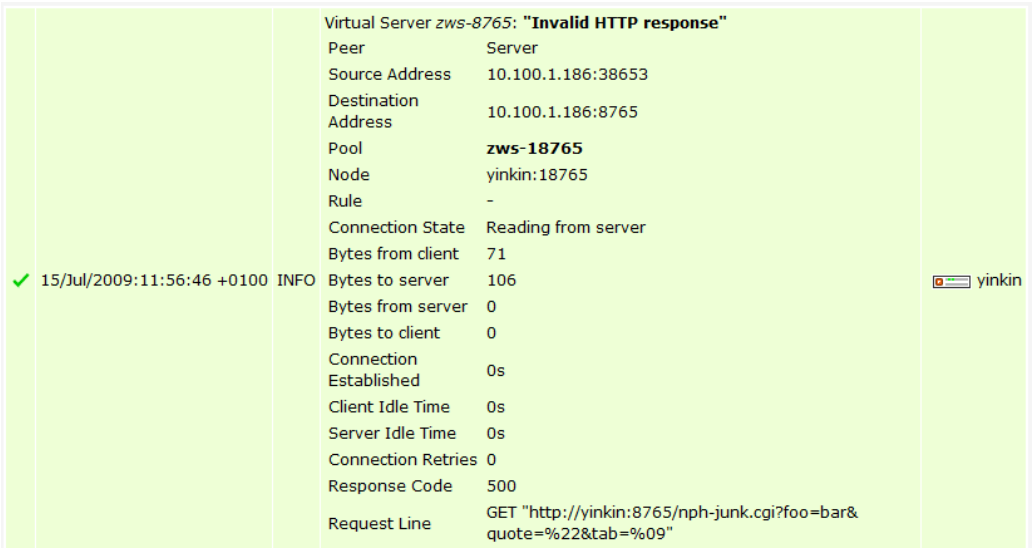


Fig 89. Connection error details

By default, connection errors are logged to the global Event Log. You can create an Event Handler that captures all Connection Failure events and writes them to a

separate log file, and bypasses the global Event Log. Refer to the *Overview* section of CHAPTER 21 for more details.

Testing Individual Nodes

To check that each back-end node in a pool can receive requests from the Traffic Manager, switch the load-balancing algorithm for the pool to **Round Robin** (see the *Load Balancing* section of CHAPTER 5). The "Round Robin" load-balancing algorithm routes requests to each node in turn; you can use the logs on your back-end servers to check that this is working for every node.

You can also configure nodes to drain connections (see the *Draining and Disabling Nodes* section of CHAPTER 5). If you have several nodes in a pool, you can temporarily stop the Traffic Manager sending traffic to some by setting them to drain. To test each node in turn, you can drain all the nodes apart from the one you wish to test.

Enable server logging on each node and verify that the logs the nodes write are consistent with the activity on the system.

Understanding Your Configuration

The configuration summary provides an overview of how requests are processed with the Traffic Manager. Requests are received by the Traffic Manager, processed by a virtual server and a pool, forwarded to a node, and then the response is processed by the pool and virtual server on the return path.

The processing path can be quite complex when a number of TrafficScript rules and the Traffic Manager capabilities are in effect.

The Configuration Summary is described in section CHAPTER 7.

If you can trace the problem under investigation to a particular time, you can use the **Audit Log** (in the **Diagnose** part of the Admin Server) and possibly the **Backup Management** (in the **System** part of the Admin Server) to identify configuration changes that may have provoked the problem.

The Audit Log records all configuration changes performed by each user of the Traffic Manager Admin Server interface. You may wish to give each authorized administrator an individual login so that you can manage their privileges and monitor their activities. User and Group configuration is described in the *User Management* section of CHAPTER 24.

Backup Management allows you to manually take snapshots of your configuration. You can compare the current configuration with a known good snapshot, and restore backups in the event of a serious configuration problem. Backup Management is described in the *System > Backup* section of CHAPTER 7.

Troubleshooting Tips

The process of diagnosing faults in a complex traffic-managed cluster is an involved one, and a systematic approach to troubleshooting is required. In the event of any problems, check through the following areas in turn to locate and diagnose the cause of the fault.

Generating Test Requests

You can test your system using the normal client software that users will employ, such as a Web browser or email client. Tools like the Live HTTP Headers extension for Mozilla browsers are very useful when inspecting the request and response flow between the client and the Traffic Manager system.

You can also use a network snooping tool such as Wireshark (http://www.wireshark.org/) to record network traffic and assemble request and response sessions.

httpclient

The `httpclient` program, included with the Traffic Manager distribution, can be used to issue HTTP requests to particular machines. You can find it at *ZEUSHOME*`/admin/bin/httpclient`.

Use the following syntax:

```
$ httpclient --hostheader=<website_name>
http://<trafficIP>/
```

You can also perform initial tests using a telnet client to perform a basic test on the service:

```
$ telnet www.mysite.com 80
Trying 62.254.209.66...
Connected to 62.254.209.66.
GET / HTTP/1.1<RETURN>
Host: www.mysite.com<RETURN>
<RETURN>
```

The openssl toolkit (http://www.openssl.org/) includes a Telnet-like client that uses the SSL protocol, for testing SSL-related problems.

zeusbench

The `zeusbench` program is a useful benchmarking tool that can be used to send large numbers of HTTP requests for load and performance testing purposes. You can find it at *`ZEUSHOME`*`/admin/bin/zeusbench`.

There are many command line options. You can run a simple load test using the following command:

```
$ zeusbench -t 30 -c 100 -k http://host/url
```

Run `zeusbench -h` for a full list of command line options.

Checking Automatic Back-End Failover

To check the Traffic Manager's automatic failover of back ends you will need at least two back-end servers configured, or there will be no machines for the Traffic Manager to fall back on. You can test failover by pulling out the network cable on one of the back-end machines; or you can manually stop the service running on the back end. Verify that nothing is then listening on that port on the back end.

If you only have one back-end server you could run two instances of the required service on different ports on the same server machine, and then manually stop one instance of the service.

Try to use your selected service through the Traffic Manager. For SMTP send an email through the server farm, or for HTTP make a Web page request. If at least one back-end server is available to fulfill your request it should succeed.

Check the event log, linked from the **Diagnose** pages, for notification that a back-end machine has failed.

Checking Automatic Front-End Failover

If you have two or more Traffic Managers, you can check that automatic front-end failover is working properly. Set up a traffic IP group spanning your Traffic Managers, and a service using this traffic IP group, such as a virtual server managing Web content on port 80. Check that you can request Web pages from this service successfully on each of the traffic IP addresses in the group. You can do this by entering the IP address rather than the DNS name in your browser.

Now click the **Services** button on the Admin Server pages. Click the **Traffic IP Groups** tab, and click **Unfold All** to view details about your traffic IP groups. This shows you which machine has raised which IP address; note that if you have more

machines than traffic IP addresses in the group, some machines will be on standby and not actively handling traffic.

Now pick a machine that has raised one of the traffic IP addresses, and pull out all the network cables on that machine. The IP address will be raised by another machine in the group; try browsing to the traffic IP address again and check that you can still receive content. You may need to refresh the page in your browser to ensure that it is not using cached content.

The *Configuring Fault-Tolerance* section of CHAPTER 6 describes how the Traffic Manager's fault tolerance works, the tests that it conducts and the decisions that it makes.

Common Problems

Did Not Become Root

The Traffic Manager needs to bind to privileged ports to balance network services like mail and HTTP (privileged ports are ports below number 1024). To do this you need to install, run the configure script and start the Traffic Manager as root.

In front-end fault tolerance mode, the Traffic Manager needs to raise and lower network interfaces. If you did not configure and start the software as root, the failover architecture does not function.

Note that the Traffic Manager runs as a non-privileged user, but it has to be configured as, and started as the root user.

For evaluation purposes, it is possible to configure and start the Traffic Manager without becoming root. However, in this case you will not be able to bind to privileged ports (see above), nor use fault tolerance.

Connection Refused

The most common configuration error for front ends is a bad DNS setup mapping the external name to the external IP addresses for your front-end machines. If you receive a "Connection Refused" message when connecting to your server through the front-end DNS name, it is likely that your DNS is not configured correctly.

To check your DNS configuration, you can use the `host` or `nslookup` commands:

```
$ host www.w3.org
www.w3.org has address 18.29.1.35
www.w3.org has address 18.7.14.127
www.w3.org has address 18.29.1.34
```

Verify that your Traffic Manager machines are listening on these IP addresses.

You can configure a virtual server to listen for traffic on these IP addresses explicitly. Then the Diagnose page will report an error if these IP addresses are not available to the Traffic Manager cluster.

Inappropriate Traffic IP Addresses Configured

If you configure a traffic IP group for front-end fault tolerance, the Traffic Manager will make a number of checks to ensure that the traffic IP addresses you select are sensible. However, you should also double-check that they are appropriate for your current network configuration. Common errors include selecting traffic IP addresses that are used elsewhere (including as permanent IP addresses on the Traffic Manager machines), IP addresses that do not lie in the subnets used by the Traffic Managers, or IP addresses that are not routable from elsewhere.

The Traffic Manager Drops Connection Before Protocol Begins

For *server first* protocols, when the Traffic Manager cannot contact any back ends for work for a particular port for server first protocols, it will have no choice other than to drop the connection with the client. If you connect to the correct port on the Traffic Manager machine (using telnet for instance), and get a successful connection (i.e. no "Connection Refused" message), but the connection drops almost immediately, the Traffic Manager is trying to talk to a back end but cannot find any.

Check the event log on the Diagnose page for messages about the status of the back-end servers.

Web Server Returns Error 400

If on testing your traffic-managed Web site you only get errors of type Error 400 Bad Request, this may be because the Web site you are trying to access has not been configured to accept any other host headers (or aliases). In order to resolve this problem you can add a rule to set the right host header:

```
http.setHeader( "Host", "www.mysite.com" );
```

Wrong Port Number Configured

Another common problem is the wrong port number being balanced. If, for instance, your Web site is running on a port other than 80, you will need to balance that port number with the HTTP protocol rather than the default. If you are running an SSL-encrypted site, you should select the HTTPS protocol rather than HTTP, or SSL-decrypt your traffic. The default port for HTTPS is 443.

Note that it is possible to balance many distinct ports with one Traffic Manager installation, provided an appropriate protocol is selected in each case.

Running Out of File Descriptors

Your Traffic Manager uses operating system resources called 'file descriptors' to manage each network connection to a client or server. There is a hard limit on the number of file descriptors it can allocate (the limit applies per process, i.e. per CPU core), set in the operating system.

Under times of very high load, such as during a benchmark, you may see an error message indicating that you are "Running out of file descriptors". In this case, you can increase the number of file descriptors that your Traffic Manager uses.

Go to the **System > Global Settings** page in the administration interface. In the **System Settings** part of the configuration, increase the value of **maxfds** to a larger value. If you are already at the maximum that your operating system allows, you should refer to your system documentation to ascertain whether this hard limit can be increased.

Running Out of Disk Space

Note: This section is applicable to appliance and cloud variants only.

Over time, your appliance can run low on disk space. For example, your system logs can become large if you have configured your Traffic Manager to produce detailed request log information. Additionally, archived software revisions (used by the Traffic Manager for roll back) might no longer be required.

The Traffic Manager warns you if disk space is running low through the Event Log and **Diagnose > Cluster Diagnosis** page. You can also view disk space usage at any time through the **System > Traffic Managers** page.

To free up disk space, click on "Free up some disk space" from the **Wizards:** drop-down menu in the top navigation bar. You can also run the wizard from the "Free Disk Space" link on the **System > Traffic Managers** page at any time, and from the **Diagnose > Cluster Diagnosis** page when a low disk space warning appears.

Important: This operation is irreversible. You should ensure you have created a backup of any files you need to keep before running the wizard. Note also that any "Technical Support Reports" you create afterwards contain only those logs generated since the wizard was run.

Getting Help

If you need to contact a technical support engineer, please include the Technical Support Report (TSR) download as an attachment to your problem report. The report download contains your recent log files, configuration, system activity and a variety of internal information.

To create, download, and manage your TSRs, click **Manage Technical Support Reports** on the **Diagnose > Technical Support** page of the Admin UI.

Use this page to choose what options are included in your generated TSR, and to access previously generated reports. The Traffic Manager stores the last 10 TSRs generated, with the option to download or delete each one. You can free disk space by downloading TSRs to an external location and then deleting them from the Traffic Manager using the check boxes provided.

You can also generate a TSR using the command line or through the SOAP API. Use these options when access to the Admin UI is interrupted.

To generate a TSR from the command line, run the following script:

```
ZEUSHOME/zxtm/bin/support-report [--all-logs] <outfile>
```

Use the optional `--all-logs` argument to bypass the default log file size download limit of 10 MB. This means that the entire log set is included in the support report, regardless of size.

CHAPTER 34 Glossary

Below is an explanation of terms used in this document. Some of these terms are used with varying meanings in computing literature; this glossary is intended only to define them within the scope of this document.

Action – An action is invoked when an event handler is triggered as the result of a particular event occurring. Possible actions include writing to log files, sending an email message, sending a SYSLOG message or SNMP trap, or running a custom action program.

Admin Server – The administration interface to the Traffic Manager family of solutions, accessed through the Web-based user-interface.

Audit log – The Traffic Manager's audit log of users' activities for the Traffic Manager. These can be viewed within the Diagnose pages.

Back-end IP address – The IP address which a Traffic Manager machine uses to communicate with a back-end server, or the IP address of a back-end server.

Back-end server – A machine which runs a service, such as a Web or mail server. A back-end server is typically within your local network, and requests handled by the Traffic Manager are passed on to it. A back-end server together with a specified port forms a node in the Traffic Manager.

Bandwidth management – The ability to monitor and control the amount of network bandwidth available to a particular service or type of request.

Catalog – A central repository for objects which you can apply to your services. There are catalogs for rules, monitors, service protection classes and various SSL items.

Certificate authority – A recognized authority which independently signs SSL certificates.

Certificate revocation list (CRL) – A list of client SSL certificates which should be considered invalid; issued by a certificate authority.

Certificate signing request (CSR) – A request created from a self-signed certificate. You can submit the CSR to a certificate authority for them to sign.

CIDR IP subnet – A Classless Inter-Domain Routing (CIDR) IP subnet includes a standard 32-bit IP address and additional information on how many bits are used for the network prefix. For example, in the CIDR IP subnet "10.13.1.48/22", the "/22" indicates that the first 22 bits are used to identify the unique network prefix and the remaining 10 bits identify

the specific host. Alternatively, the Traffic Manager allows to specify a CIDR IP subnet using a subnet mask "10.13.1.48/255.255.255.0" or explicitly "10.13.1.".

Client certificate – An SSL certificate held by a client, granting the client access to a restricted site.

Cluster – A group of Traffic Managers which share the same configuration.

configure script – The script which must be run after installing and before starting the Traffic Manager. It deals with fundamental settings such as passwords and specifying whether the Traffic Manager should stand alone or join an existing cluster.

Connection management – Settings for managing the connection between a remote client and a virtual server, or between a pool and its nodes.

Cookie - A small item of data given to a client by a server that is stored either on the client's file system or in the browser client session. The client stores the data and provides it to the server on subsequent requests. Cookies are used to track client data such as session persistence maps.

Denial of Service attack – A malicious attempt to prevent legitimate use of a service, often by flooding a network or disrupting connectivity between machines.

Diagnose pages – The section of the Admin Server which helps you diagnose and fix any faults with your Traffic Manager setup.

Disabled node – A node in a pool that is not used; it is not monitored, and no traffic is sent to it. Disabling a node is an alternative to removing the node from the pool, but disabling it allows it to be reinstated more easily.

Distributed Denial of Service attack – A Denial of Service attack which comes from a group of machines; these machines have often been compromised by worms or viruses.

DMZ – The demilitarized zone for a firewall. This zone sits between the internal network and the Internet, and often holds your externally available servers.

Draining node – A node in a pool which is being sent no new connections. When all existing connections and sessions to this node have expired, it can be removed from the pool safely.

Error file – A file sent to the client when no nodes are available to respond to its request. Error files are set up on a per-pool basis.

Event – An event is raised when a particular change or occurrence takes place, such as an IP Address transfer or a node failure or recovery. All events are logged to the global Event Log.

Event Handler – An event hander configures the actions that the Traffic Manager should take when an event in the associated Event Type occurs.

Event Log – The Traffic Manager's log of events within the Traffic Manager. These are graded as INFO, WARN, SERIOUS, FATAL etc, and can be viewed within the Diagnose pages.

Event Type – A set of events, grouped for administrative convenience. When any event in a particular event type occurs, the Traffic Manager can run the actions associated with that event type.

Failover – The act of taking on a failed machine's load by other machines in the cluster.

Failure pool – If all the nodes in a pool should fail, requests will be sent to its failure pool, if one is configured.

Fault tolerance – The ability of a cluster or pool to cope with failure of one or more of its servers, without interruption to service.

Front-end IP address – The permanent, externally available IP address of a Traffic Manager.

Front-end server – A server which is contactable from the Internet, such as a Traffic Manager.

Health monitoring – The process by which the Traffic Manager tracks the health of Traffic Managers and back ends, so that requests will not be sent to a failed machine.

IPv6 - A network layer protocol that is expected to supersede the current protocol, IPv4. Its features include a bigger range of addresses, higher security and further advantages.

Java extension - Java Extensions can increase ('extend') the capabilities of TrafficScript. A TrafficScript rule (or RuleBuilder rule) can invoke a Java Extension to process a request or response, and the Java run-time environment provides a much more powerful set of programming language tools and libraries than TrafficScript does alone.

Load balancing – The distribution of requests among a number of back-end servers, so that every request is dealt with as quickly as possible.

MD5 - In cryptography, MD5 (Message-Digest algorithm 5) is a widely used cryptographic hash function with a 128-bit hash value. As an Internet standard (RFC 1321), MD5 has been employed in a wide variety of security applications, and is also commonly used to check the integrity of files.

MIME type - Multipart Internet Mail Extension, a name which identifies the type of data being transferred, e.g. 'application/pdf' for PDF files. This standard was first used in email applications, and has been widely adopted by other Internet technologies such as Web servers.

Monitor – A regular test which the Traffic Manager uses to track the health of a pool.

Name server – A server that implements a name service protocol. For example, a Domain Name System (DNS) server might translate the domain name www.example.com to the IP address 208.70.196.59.

NAT – Network address translation (NAT, also known as network masquerading) involves rewriting the source or destination addresses of IP packets as they pass through a router or firewall. Most systems using NAT do so in order to enable multiple hosts on a private network to access the Internet using a single public IP address.

Node – A back-end server with an associated port, which receives requests sent to it by the Traffic Manager and responds with the requested content.

PEM –Private Enhanced Mail. The .pem file extension is used for Base 64-encoded X.509 keys and certificates. These are printable text, and start with a line of the form: ----- BEGIN CERTIFICATE -----

PKCS#11 – The standard used to communicate with secure hardware.

Pool – A logical group of nodes. When a virtual server assigns requests to a pool, it load-balances them across the nodes.

Real Time Streaming Protocol (RTSP) - An application-level protocol that enables on-demand delivery of real-time data, such as audio and video formats, using the TCP and UDP protocols.

Real-time Transport Protocol (RTP) – A UDP packet format for delivering audio and video files, which is commonly used alongside RTSP to provide the delivery of the streaming data.

Request Rule – A TrafficScript or RuleBuilder rule that controls how the Traffic Manager should handle a request. Rules are invoked by a virtual server, and can inspect, manipulate, rewrite and route requests; they

can also control how other capabilities in the Traffic Manager are used to manage the request.

Response Rule – the counterpart of request rules, response rules control how the Traffic Manager should handle a response.

RuleBuilder - A visual editor for creating rules, which allows you to select conditions on the request to be examined, and actions that follow if any or all of these conditions are met.

Self-signed SSL certificate – An SSL certificate signed by its owner, rather than a certificate authority. These are useful for security testing but should not be used for live sites. You can issue a certificate signing request to convert your certificate to one signed by a certificate authority.

Server certificate – An SSL certificate used to identify an SSL service (such as a Web site), and containing the public key necessary to set up the encrypted transaction.

Service – Common Internet services include Web, mail, DNS and applications such as .NET and SOAP. They are provided, publicly or locally, via an IP or DNS address and port combination.

Service level monitoring – The ability to monitor back-end server response times, compare them to a conformance value and react to changes, including dynamically adjusting the resources available.

Service protection – The protection of your Internet services from attacks such as denial of service (DoS) and distributed denial of service (DDoS).

Service protection class – A group of service protection settings you define. The class is held in the Service Protection Catalog and can be applied to any virtual server.

Session Initiation Protocol (SIP) - A protocol used in Internet for conferencing, telephony, presence, events and instant messaging.

Session persistence – A process by which requests belonging to the same user session are always sent to the same node. Sessions can be identified by several methods including IP addresses, cookies or SSL session IDs.

SNMP – Simple Network Management Protocol, an open standard for network monitoring and management. It allows you to monitor devices on a network and gather performance data.

SOAP - Simple Object Access Protocol, an XML-based standard for Web services messages.

SSL – Secure Sockets Layer, a protocol which provides encrypted communications over a network. Requests can be decrypted and re-encrypted within the Traffic Manager.

Tarball –A file in the tar file format, a standard type of archive file format. It is used widely to archive and unarchive files while preserving file system information such as user and group permissions, dates, and directory structures.

Traffic IP address – An IP address you use to publish an Internet service, usually linked to your public DNS entry.

Traffic IP group – A logical group containing several traffic IP addresses, and spanning some or all of your Traffic Managers. These Traffic Managers negotiate to ensure that the IP addresses in the group are always available.

Traffic manager – A front-end server running the Traffic Manager and managing your services.

Traffic cluster – The ability to deploy unlimited numbers of active and passive Traffic Managers, providing resilience to multiple failures and the ability to scale the Traffic Manager cluster as the need arises. Also known as N+M Redundancy.

TrafficScript – The scripting language used to write rules for the Traffic Manager range of solutions.

Vector – The means or vehicle by which a Denial of Service or Distributed Denial of Service attack is made.

Virtual server – An interface between the Traffic Manager and the Internet. A virtual server manages one of your services. It receives a request from the Internet and, possibly after SSL-decrypting it and applying rules, assigns it to a pool.

XML – Extensible Markup Language, used for storing and exchanging structured data.

XPath – A language for addressing parts of an XML document. It can be used in conjunction with TrafficScript rules to make decisions based on the document's content.

ZEUSHOME – The location in which you install the Traffic Manager. The default is /usr/local/zeus for Traffic Manager software variants, and /opt/zeus for Traffic Manager virtual appliances/cloud instances.

CHAPTER 35 Software License Acknowledgements

License for the Berkeley DB Code (Version 1.85)

The following license refers to the libdb code that the Traffic Manager is linked against. The source for this code can be found at:

http://www.sleepycat.com/update/snapshot/db.1.85.tar.gz

Copyright (c) 1991, 1993 The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the University of California, Berkeley and its contributors.

Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

RSA PKCS11

License to copy and use this software is granted provided that it is identified as "RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki)" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki)" in all material mentioning or referencing the derived work.

License for the OpenLDAP Code, Version 2.4.23

The following license refers to the libldap code that the Traffic Manager is linked against. The source for this code can be found at:

ftp://ftp.openldap.org/pub/OpenLDAP/openldap-stable/openldap-stable-20100719.tgz

The OpenLDAP Public License

Version 2.8, 17 August 2003

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions in source form must retain copyright statements and notices,

2. Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and

3. Redistributions must contain a verbatim copy of this document.

The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.

THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,

PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The names of the authors and copyright holders must not be used in advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

PCRE License

The following license refers to the pcre code that the Traffic Manager is linked against.

Copyright (c) 1997-2008 University of Cambridge

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the University of Cambridge nor the name of Google Inc. nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Libnet License

The following license refers to the libnet code that the Traffic Manager is linked against (on Solaris and FreeBSD versions).

libnet 1.1.x

Copyright (c) 1998 - 2002 Mike D. Schiffman <mike@infonexus.com>

http://www.packetfactory.net/libnet

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF

THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for Yahoo! UI Library

Copyright (c) 2009, Yahoo! Inc. All rights reserved.

Redistribution and use of this software in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Yahoo! Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission of Yahoo! Inc.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for ssleay Cryptographic Library

The following license refers to an implementation of the SHA-1 and MD5 algorithms used in Brocade Virtual Traffic Manager:

Copyright (C) 1997 Eric Young (eay@cryptsoft.com). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)". The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for libxml2 and libxslt

The following license refers to the XML parser and toolkit libraries used in Brocade Virtual Traffic Manager, as licensed under the MIT License:

libxml2 2.9.1
libxslt 1.1.28

Copyright (c) 1998-2012 Daniel Veillard

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.

License for the Java Servlet API

The Java Servlet API used in Brocade Virtual Traffic Manager contains code from the Apache Jakarta Project, and is licensed under the ASF license version 2.0:

Copyright 1999-2007, The Apache Software Foundation

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

License for the Expat XML Parser

The following license refers to the expat XML parser library used in Brocade Virtual Traffic Manager:

expat 2.0.0

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd
and Clark Cooper
Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY
CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT,

TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for MooTools

Brocade Virtual Traffic Manager contains code from the MooTools JavaScript framework, licensed under the open source MIT license:

MooTools 1.2

Copyright (c) 2006-2009 Valerio Proietti

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Licenses for OpenLayers

Brocade Virtual Traffic Manager contains code from MetaCarta, Rico and XMLHttpRequest.js. The following copyright notices apply:

Copyright (c) 2005-2008 MetaCarta, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted (subject to the limitations in the disclaimer below) provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of MetaCarta, Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

NO EXPRESS OR IMPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE GRANTED BY THIS LICENSE. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Contains portions of Rico <http://openrico.org/>

Copyright 2005 Sabre Airline Solutions

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Contains XMLHttpRequest.js <http://code.google.com/p/xmlhttprequest/>

Copyright 2007 Sergey Ilinsky (http://www.ilinsky.com)

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at:

http://www.apache.org/licenses/LICENSE-2.0

License for rsync

Brocade Virtual Traffic Manager contains code from rsync – a fast remote file copy program. The following copyright notice applies:

Version: 2.6.9

License summary: GPL v2

Copyright:

rsync was originally written by Andrew Tridgell and has been improved by many developers around the world. rsync may be used, modified and redistributed only under the terms of the GNU General Public License, found at:

http://www.fsf.org/licenses/old-licenses/gpl-2.0.html

License for mod_imap.c

Brocade Virtual Traffic Manager contains code from mod_imap.c, which originates from the Apache HTTP Server (http://httpd.apache.org) and is licensed under version 1.1 of the Apache Software License. The following copyright notice applies:

The Apache Software License, Version 1.1

Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:

"This product includes software developed by the Apache Software Foundation (http://www.apache.org/)."

Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.

4. The names "Apache" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.

5. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR

BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for Antlr and libantlr

Brocade Virtual Traffic Manager contains code from Antlr and libantlr – a JavaScript lexer generator and runtime library. The following copyright notice applies:

Version: 3.4

License summary: BSD

Copyright:

Copyright (c) 2005-2009 Jim Idle, Temporal Wave LLC

http://www.temporal-wave.com

http://www.linkedin.com/in/jimidle

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA,

OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for es3-grammar

Brocade Virtual Traffic Manager contains code generated using es3-grammar – a complete ECMAScript 3 grammar for ANTLR 3. The following copyright notice applies:

Version: N/A

License summary: BSD

Copyright:

Copyright (c) 2008-2010, Xebic Research B.V. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY

OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for jsoncpp

Brocade Virtual Traffic Manager contains code from jsoncpp – a JSON library used for parsing config files. The following copyright notice applies:

Version: 0.5.0

License summary: MIT

Copyright:

Copyright (c) 2007-2010 Baptiste Lepilleur

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for libjpeg

Brocade Virtual Traffic Manager contains code from libjpeg – a JPEG manipulation library. The following copyright notice applies:

Version: 8c

License summary: IJG license

Copyright:

The authors make NO WARRANTY or representation, either express or implied, with respect to this software, its quality, accuracy, merchantability, or fitness for a particular purpose. This software is provided "AS IS", and you, its user, assume the entire risk as to its quality and accuracy.

This software is copyright (C) 1991-2011, Thomas G. Lane, Guido Vollbeding. All Rights Reserved except as specified below.

Permission is hereby granted to use, copy, modify, and distribute this software (or portions thereof) for any purpose, without fee, subject to these conditions:

(1) If any part of the source code for this software is distributed, then this README file must be included, with this copyright and no-warranty notice unaltered; and any additions, deletions, or changes to the original files must be clearly indicated in accompanying documentation.

(2) If only executable code is distributed, then the accompanying documentation must state that "this software is based in part on the work of the Independent JPEG Group".

(3) Permission for use of this software is granted only if the user accepts full responsibility for any undesirable consequences; the authors accept NO LIABILITY for damages of any kind.

These conditions apply to any software derived from or based on the IJG code, not just to the unmodified library. If you use our work, you ought to acknowledge us.

Permission is NOT granted for the use of any IJG author's name or company name in advertising or publicity relating to this software or products derived from it. This software may be referred to only as "the Independent JPEG Group's software".

We specifically permit and encourage the use of this software as the basis of commercial products, provided that all warranty or liability claims are assumed by the product vendor.

ansi2knr.c is included in this distribution by permission of L. Peter Deutsch, sole proprietor of its copyright holder, Aladdin Enterprises of Menlo Park, CA. ansi2knr.c is NOT covered by the above copyright and conditions, but instead by the usual distribution terms of the Free Software Foundation; principally, that you must include source code if you redistribute it. (See the file ansi2knr.c for full details.) However, since ansi2knr.c is not needed as part of any program generated from the IJG code, this does not limit you more than the foregoing paragraphs do.

The Unix configuration script "configure" was produced with GNU Autoconf. It is copyright by the Free Software Foundation but is freely distributable. The same holds for its supporting scripts (config.guess, config.sub, ltmain.sh). Another

support script, install-sh, is copyright by X Consortium but is also freely distributable.

The IJG distribution formerly included code to read and write GIF files. To avoid entanglement with the Unisys LZW patent, GIF reading support has been removed altogether, and the GIF writer has been simplified to produce "uncompressed GIFs". This technique does not use the LZW algorithm; the resulting GIF files are larger than usual, but are readable by all standard GIF decoders.

We are required to state that

"The Graphics Interchange Format(c) is the Copyright property of CompuServe Incorporated. GIF(sm) is a Service Mark property of CompuServe Incorporated."

License for libunwind

Brocade Virtual Traffic Manager contains code from libunwind – a stack unwinding library. The following copyright notice applies:

Version: 1.0.1

License summary: MIT

Copyright:

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for the Perl JSON Library

Brocade Virtual Traffic Manager contains code from the Perl JSON library – a Perl module for JSON (JavaScript Object Notation) encoding and decoding. The following copyright notice applies:

Version: 2.53

License summary: Perl Artistic License v.1

Copyright:

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions:

> "Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.
>
> "Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.
>
> "Copyright Holder" is whoever is named in the copyright or copyrights for the package.
>
> "You" is you, if you're thinking about copying or distributing this Package.
>
> "Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)
>
> "Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.

2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.

3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:

> a) place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
>
> b) use the modified Package only within your corporation or organization.
>
> c) rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.
>
> d) make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

> a) distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.
>
> b) accompany the distribution with the machine-readable source of the Package with your modifications.
>
> c) give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
>
> d) make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours

(by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package via the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

License for OpenSSL

Brocade Virtual Traffic Manager contains code from the OpenSSL library – a cryptography and SSL/TLS toolkit. The following copyright notice applies:

Version: 1.0.2d

License summary: BSD-style

Copyright notice:

LICENSE ISSUES

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL please contact openssl-core@openssl.org.

OpenSSL License

Copyright (c) 1998-2011 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)"

4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.

6. Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL

DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related :-).

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

License for WebP

Brocade Virtual Traffic Manager contains code from WebP – an image format with corresponding conversion library. The following copyright notice applies:

Version: 0.3.1

License summary: BSD-style

Copyright notice:

Copyright (c) 2010, Google Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of Google nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for Flex

Brocade Virtual Traffic Manager contains code generated through Flex – a fast lexical analyzer tool. The following copyright notice applies:

Version: 2.5.37

License summary: BSD-style

Copyright notice:

Flex carries the copyright used for BSD software, slightly modified because it originated at the Lawrence Berkeley (not Livermore!) Laboratory, which operates under a contract with the Department of Energy:

Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006, 2007 The Flex Project.

Copyright (c) 1990, 1997 The Regents of the University of California.

All rights reserved.

This code is derived from software contributed to Berkeley byVern Paxson.

The United States Government has rights in this work pursuant to contract no. DE-AC03-76SF00098 between the United States Department of Energy and the University of California.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

This basically says "do whatever you please with this software except remove this notice or take advantage of the University's (or the flex authors') name".

Note that the "flex.skl" scanner skeleton carries no copyright notice. You are free to do whatever you please with scanners generated using flex; for them, you are not even bound by the above copyright.

License for CryptoJS

Brocade Virtual Traffic Manager contains code from CryptoJS - a collection of standard and secure cryptographic algorithms implemented in JavaScript. The following copyright notice applies:

Version: 3.1.2

License summary: BSD

Copyright notice:

Copyright (c) 2009-2013 Jeff Mott

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify,

merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for zlib.js

Brocade Virtual Traffic Manager contains code from zlib.js – a JavaScript Zlib library. The following copyright notice applies:

Version: 0.2.0

License summary: MIT

Copyright notice:

Copyright (c) 2012 imaya

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING

FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for XML::Twig

Brocade Virtual Traffic Manager contains code from XML::Twig – a Perl library used for processing XML data. The following copyright notice applies:

Version: 3.48

License summary: Perl 5 License (Artistic 1 & GPL 1)

Copyright notice:

COPYRIGHT

Copyright (c) 1999-2012, Michel Rodriguez. All Rights Reserved.

This library is free software; you can redistribute it and/or modify

it under the same terms as Perl itself.

License for MIT Kerberos

Brocade Virtual Traffic Manager contains code from MIT Kerberos – a network authentication protocol. The following copyright notice applies:

Version: 1.12.1

License summary: MIT

Copyright notice:

Copyright © 1985-2014 by the Massachusetts Institute of Technology.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Downloading of this software may constitute an export of cryptographic software from the United States of America that is subject to the United States Export Administration Regulations (EAR), 15 CFR 730-774. Additional laws or regulations may apply. It is the responsibility of the person or entity contemplating export to comply with all applicable export laws and regulations, including obtaining any required license from the U.S. government.

The U.S. government prohibits export of encryption source code to certain countries and individuals, including, but not limited to, the countries of Cuba, Iran, North Korea, Sudan, Syria, and residents and nationals of those countries.

Documentation components of this software distribution are licensed under a Creative Commons Attribution-ShareAlike 3.0 Unported License. (http://creativecommons.org/licenses/by-sa/3.0/)

Individual source code files are copyright MIT, Cygnus Support, Novell, OpenVision Technologies, Oracle, Red Hat, Sun Microsystems, FundsXpress, and others.

Project Athena, Athena, Athena MUSE, Discuss, Hesiod, Kerberos, Moira, and Zephyr are trademarks of the Massachusetts Institute of Technology (MIT). No commercial use of these trademarks may be made without prior written permission of MIT.

"Commercial use" means use of a name in a product or other for-profit manner. It does NOT prevent a commercial firm from referring to the MIT trademarks in order to convey information (although in doing so, recognition of their trademark status should be given).

The following copyright and permission notice applies to the OpenVision Kerberos Administration system located in kadmin/create, kadmin/dbutil, kadmin/passwd, kadmin/server, lib/kadm5, and portions of lib/rpc:

Copyright, OpenVision Technologies, Inc., 1993-1996, All Rights Reserved

WARNING: Retrieving the OpenVision Kerberos Administration system source code, as described below, indicates your acceptance of the following terms. If you do not agree to the following terms, do not retrieve the OpenVision Kerberos administration system.

You may freely use and distribute the Source Code and Object Code compiled from it, with or without modification, but this Source Code is provided to you "AS IS" EXCLUSIVE OF ANY WARRANTY, INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY OTHER WARRANTY, WHETHER EXPRESS OR IMPLIED. IN NO EVENT WILL OPENVISION HAVE ANY LIABILITY FOR ANY LOST PROFITS, LOSS OF DATA OR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THIS AGREEMENT, INCLUDING, WITHOUT LIMITATION, THOSE RESULTING FROM THE USE OF THE SOURCE CODE, OR THE FAILURE OF THE SOURCE CODE TO PERFORM, OR FOR ANY OTHER REASON.

OpenVision retains all copyrights in the donated Source Code. OpenVision also retains copyright to derivative works of the Source Code, whether created by OpenVision or by a third party. The OpenVision copyright notice must be preserved if derivative works are made based on the donated Source Code.

OpenVision Technologies, Inc. has donated this Kerberos Administration system to MIT for inclusion in the standard Kerberos 5 distribution. This donation underscores our commitment to continuing Kerberos technology development and our gratitude for the valuable work which has been performed by MIT and the Kerberos community.

Portions contributed by Matt Crawford crawdad@fnal.gov were work performed at Fermi National Accelerator Laboratory, which is operated by Universities Research Association, Inc., under contract DE-AC02-76CHO3000 with the U.S. Department of Energy.

Portions of src/lib/crypto have the following copyright:

Copyright © 1998 by the FundsXpress, INC.

All rights reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of FundsXpress. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. FundsXpress makes no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

THIS SOFTWARE IS PROVIDED “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The implementation of the AES encryption algorithm in src/lib/crypto/builtin/aes has the following copyright:

Copyright © 2001, Dr Brian Gladman brg@gladman.uk.net, Worcester, UK.

All rights reserved.

LICENSE TERMS

The free distribution and use of this software in both source and binary form is allowed (with or without changes) provided that:

distributions of this source code include the above copyright notice, this list of conditions and the following disclaimer;

distributions in binary form include the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other associated materials;

the copyright holder’s name is not used to endorse products built using this software without specific written permission.

DISCLAIMER

This software is provided ‘as is’ with no explcit or implied warranties in respect of any properties, including, but not limited to, correctness and fitness for purpose.

Portions contributed by Red Hat, including the pre-authentication plug-in framework and the NSS crypto implementation, contain the following copyright:

Copyright © 2006 Red Hat, Inc.

Portions copyright © 2006 Massachusetts Institute of Technology

All Rights Reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Red Hat, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The bundled verto source code is subject to the following license:

Copyright 2011 Red Hat, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT

HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The implementations of GSSAPI mechglue in GSSAPI-SPNEGO in src/lib/gssapi, including the following files:

lib/gssapi/generic/gssapi_err_generic.et

lib/gssapi/mechglue/g_accept_sec_context.c

lib/gssapi/mechglue/g_acquire_cred.c

lib/gssapi/mechglue/g_canon_name.c

lib/gssapi/mechglue/g_compare_name.c

lib/gssapi/mechglue/g_context_time.c

lib/gssapi/mechglue/g_delete_sec_context.c

lib/gssapi/mechglue/g_dsp_name.c

lib/gssapi/mechglue/g_dsp_status.c

lib/gssapi/mechglue/g_dup_name.c

lib/gssapi/mechglue/g_exp_sec_context.c

lib/gssapi/mechglue/g_export_name.c

lib/gssapi/mechglue/g_glue.c

lib/gssapi/mechglue/g_imp_name.c

lib/gssapi/mechglue/g_imp_sec_context.c

lib/gssapi/mechglue/g_init_sec_context.c

lib/gssapi/mechglue/g_initialize.c

lib/gssapi/mechglue/g_inquire_context.c

lib/gssapi/mechglue/g_inquire_cred.c

lib/gssapi/mechglue/g_inquire_names.c

lib/gssapi/mechglue/g_process_context.c

lib/gssapi/mechglue/g_rel_buffer.c

lib/gssapi/mechglue/g_rel_cred.c

lib/gssapi/mechglue/g_rel_name.c

lib/gssapi/mechglue/g_rel_oid_set.c

lib/gssapi/mechglue/g_seal.c

lib/gssapi/mechglue/g_sign.c

lib/gssapi/mechglue/g_store_cred.c

lib/gssapi/mechglue/g_unseal.c

lib/gssapi/mechglue/g_userok.c

lib/gssapi/mechglue/g_utils.c

lib/gssapi/mechglue/g_verify.c

lib/gssapi/mechglue/gssd_pname_to_uid.c

lib/gssapi/mechglue/mglueP.h

lib/gssapi/mechglue/oid_ops.c

lib/gssapi/spnego/gssapiP_spnego.h

lib/gssapi/spnego/spnego_mech.c

and the initial implementation of incremental propagation, including the following new or changed files:

include/iprop_hdr.h

kadmin/server/ipropd_svc.c

lib/kdb/iprop.x

lib/kdb/kdb_convert.c

lib/kdb/kdb_log.c

lib/kdb/kdb_log.h

lib/krb5/error_tables/kdb5_err.et

slave/kpropd_rpc.c

slave/kproplog.c

are subject to the following license:

Copyright © 2004 Sun Microsystems, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Kerberos V5 includes documentation and software developed at the University of California at Berkeley, which includes this copyright notice:

Copyright © 1983 Regents of the University of California.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,

INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions contributed by Novell, Inc., including the LDAP database backend, are subject to the following license:

Copyright © 2004-2005, Novell, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

The copyright holder's name is not used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions funded by Sandia National Laboratory and developed by the University of Michigan's Center for Information Technology Integration, including the PKINIT implementation, are subject to the following license:

COPYRIGHT © 2006-2007

THE REGENTS OF THE UNIVERSITY OF MICHIGAN

ALL RIGHTS RESERVED

Permission is granted to use, copy, create derivative works and redistribute this software and such derivative works for any purpose, so long as the name of The University of Michigan is not used in any advertising or publicity pertaining to the use of distribution of this software without specific, written prior authorization. If the above copyright notice or any other identification of the University of Michigan is included in any copy of any portion of this software, then the disclaimer below must also be included.

THIS SOFTWARE IS PROVIDED AS IS, WITHOUT REPRESENTATION FROM THE UNIVERSITY OF MICHIGAN AS TO ITS FITNESS FOR ANY PURPOSE, AND WITHOUT WARRANTY BY THE UNIVERSITY OF MICHIGAN OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE REGENTS OF THE UNIVERSITY OF MICHIGAN SHALL NOT BE LIABLE FOR ANY DAMAGES, INCLUDING SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WITH RESPECT TO ANY CLAIM ARISING OUT OF OR IN CONNECTION WITH THE USE OF THE SOFTWARE, EVEN IF IT HAS BEEN OR IS HEREAFTER ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The pkcs11.h file included in the PKINIT code has the following license:

Copyright 2006 g10 Code GmbH

Copyright 2006 Andreas Jellinghaus

This file is free software; as a special exception the author gives unlimited permission to copy and/or distribute it, with or without modifications, as long as this notice is preserved.

This file is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY, to the extent permitted by law; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Portions contributed by Apple Inc. are subject to the following license:

Copyright 2004-2008 Apple Inc. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Apple Inc. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Apple Inc. makes no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

THIS SOFTWARE IS PROVIDED “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The implementations of UTF-8 string handling in src/util/support and src/lib/krb5/unicode are subject to the following copyright and permission notice:

The OpenLDAP Public License

Version 2.8, 17 August 2003

Redistribution and use of this software and associated documentation (“Software”), with or without modification, are permitted provided that the following conditions are met:

Redistributions in source form must retain copyright statements and notices,

Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and

Redistributions must contain a verbatim copy of this document.

The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.

THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS “AS IS” AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA,

OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The names of the authors and copyright holders must not be used in advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

Marked test programs in src/lib/krb5/krb have the following copyright:

Copyright © 2006 Kungliga Tekniska Högskola

(Royal Institute of Technology, Stockholm, Sweden).

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of KTH nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY KTH AND ITS CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL KTH OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER

CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions of the RPC implementation in src/lib/rpc and src/include/gssrpc have the following copyright and permission notice:

Copyright © 2010, Oracle America, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the “Oracle America, Inc.” nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2006,2007,2009 NTT (Nippon Telegraph and Telephone Corporation). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY NTT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL NTT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright 2000 by Carnegie Mellon University

All Rights Reserved

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Carnegie Mellon University not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

CARNEGIE MELLON UNIVERSITY DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL CARNEGIE MELLON UNIVERSITY BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Copyright © 2002 Naval Research Laboratory (NRL/CCS)

Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof.

NRL ALLOWS FREE USE OF THIS SOFTWARE IN ITS "AS IS" CONDITION AND DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.

Portions extracted from Internet RFCs have the following copyright notice:

Copyright © The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright © 1991, 1992, 1994 by Cygnus Support.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. Cygnus Support makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

Copyright © 2006 Secure Endpoints Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,

WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Portions of the implementation of the Fortuna-like PRNG are subject to the following notice:

Copyright © 2005 Marko Kreen

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 1994 by the University of Southern California

EXPORT OF THIS SOFTWARE from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to copy, modify, and distribute this software and its documentation in source and binary forms is hereby granted, provided that any documentation or other materials related to such distribution or

use acknowledge that the software was developed by the University of Southern California.

DISCLAIMER OF WARRANTY. THIS SOFTWARE IS PROVIDED “AS IS”. The University of Southern California MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. By way of example, but not limitation, the University of Southern California MAKES NO REPRESENTATIONS OR WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE. The University of Southern California shall not be held liable for any liability nor for any direct, indirect, or consequential damages with respect to any claim by the user or distributor of the ksu software.

Copyright © 1995

The President and Fellows of Harvard University

This code is derived from software contributed to Harvard by Jeremy Rassen.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the University of California, Berkeley and its contributors.

Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF

LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2008 by the Massachusetts Institute of Technology.

Copyright 1995 by Richard P. Basch. All Rights Reserved.

Copyright 1995 by Lehman Brothers, Inc. All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Richard P. Basch, Lehman Brothers and M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Richard P. Basch, Lehman Brothers and M.I.T. make no representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

The following notice applies to src/lib/krb5/krb/strptime.c and src/include/k5-queue.h.

Copyright © 1997, 1998 The NetBSD Foundation, Inc.

All rights reserved.

This code was contributed to The NetBSD Foundation by Klaus Klein.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the NetBSD Foundation, Inc. and its contributors.

Neither the name of The NetBSD Foundation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following notice applies to Unicode library files in src/lib/krb5/unicode:

Copyright 1997, 1998, 1999 Computing Research Labs,

New Mexico State University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE COMPUTING RESEARCH LAB OR NEW MEXICO STATE UNIVERSITY BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The following notice applies to src/util/support/strlcpy.c:

Copyright © 1998 Todd C. Miller Todd.Miller@courtesan.com

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED “AS IS” AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

The following notice applies to src/util/profile/argv_parse.c and src/util/profile/argv_parse.h:

Copyright 1999 by Theodore Ts’o.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies. THE SOFTWARE IS PROVIDED “AS IS” AND THEODORE TS’O (THE AUTHOR) DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE. (Isn’t it sick that the U.S. culture of lawsuit-happy lawyers requires this kind of disclaimer?)

The following notice applies to SWIG-generated code in src/util/profile/profile_tcl.c:

Copyright © 1999-2000, The University of Chicago

This file may be freely redistributed without license or fee provided this copyright message remains intact.

The following notice applies to portiions of src/lib/rpc and src/include/gssrpc:

Copyright © 2000 The Regents of the University of Michigan. All rights reserved.

Copyright © 2000 Dug Song dugsong@UMICH.EDU. All rights reserved, all wrongs reversed.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Implementations of the MD4 algorithm are subject to the following notice:

Copyright © 1990, RSA Data Security, Inc. All rights reserved.

License to copy and use this software is granted provided that it is identified as the “RSA Data Security, Inc. MD4 Message Digest Algorithm” in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as “derived from the RSA Data Security, Inc. MD4 Message Digest Algorithm” in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided “as is” without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

Implementations of the MD5 algorithm are subject to the following notice:

Copyright © 1990, RSA Data Security, Inc. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message- Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

The following notice applies to src/lib/crypto/crypto_tests/t_mddriver.c:

Copyright © 1990-2, RSA Data Security, Inc. Created 1990. All rights reserved.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.

Portions of src/lib/krb5 are subject to the following notice:

Copyright © 1994 CyberSAFE Corporation.

Copyright 1990,1991,2007,2008 by the Massachusetts Institute of Technology.

All Rights Reserved.

Export of this software from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation,

and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. Furthermore if you modify this software you must label your software as modified software and not distribute it in such a fashion that it might be confused with the original M.I.T. software. Neither M.I.T., the Open Computing Security Group, nor CyberSAFE Corporation make any representations about the suitability of this software for any purpose. It is provided “as is” without express or implied warranty.

Portions contributed by PADL Software are subject to the following license:

Copyright (c) 2011, PADL Software Pty Ltd. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of PADL Software nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY PADL SOFTWARE AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL PADL SOFTWARE OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The bundled libev source code is subject to the following license:

All files in libev are Copyright (C)2007,2008,2009 Marc Alexander Lehmann.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Alternatively, the contents of this package may be used under the terms of the GNU General Public License (“GPL”) version 2 or any later version, in which case the provisions of the GPL are applicable instead of the above. If you wish to allow the use of your version of this package only under the terms of the GPL and not to allow others to use your version of this file under the BSD license, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the GPL in this and the other files of this package. If you do not delete the provisions above, a recipient may use your version of this file under either the BSD or the GPL.

Files copied from the Intel AESNI Sample Library are subject to the following license:

Copyright © 2010, Intel Corporation All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Intel Corporation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for Libedit

Brocade Virtual Traffic Manager contains code from Libedit – a command line editor library. The following copyright notice applies:

Version: 0.3

License summary: BSD 3-clause "New" or "Revised" License

Copyright notice:

Copyright (c) 1997 The NetBSD Foundation, Inc. All rights reserved.

This code is derived from software contributed to The NetBSD Foundation by Jaromir Dolecek.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   This product includes software developed by the NetBSD Foundation, Inc. and its contributors.

4. Neither the name of The NetBSD Foundation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE NETBSD FOUNDATION, INC. AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FOUNDATION OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 1992, 1993 The Regents of the University of California. All rights reserved.

This code is derived from software contributed to Berkeley by Christos Zoulas of Cornell University.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

License for ZebOS

Brocade Virtual Traffic Manager contains code from the ZebOS routing suite, distributed under license from IP Infusion, Inc. The ZebOS routing suite is Copyright (C) 2001-2014 IP Infusion, Inc. All Rights Reserved.

License for Curl

Brocade Virtual Traffic Manager contains code from Curl – a command line tool and library for transferring data. The following copyright notice applies:

Version: 7.42.1

License summary: MIT

Copyright notice:

Copyright (c) 1996 - 2015, Daniel Stenberg, <daniel@haxx.se>.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

License for Jansson

Brocade Virtual Traffic Manager contains code from Jansson – a JSON support library. The following copyright notice applies:

Version: 2.7

License summary: MIT

Copyright notice:

Copyright (c) 2009-2014 Petri Lehtinen <petri@digip.org>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT

HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License for Digest::SHA

Brocade Virtual Traffic Manager contains code from Digest::SHA – a Perl library used to calculate SHA message digests. The following copyright notice applies:

Version: 5.95

License summary: Perl Artistic License

Copyright notice:

Copyright (C) 2003-2015 Mark Shelor

This library is free software; you can redistribute it and/or modify it under the same terms as Perl itself. See the Perl Artistic License for details:

http://search.cpan.org/perldoc?perlartistic

License for Sys::SysLog

Brocade Virtual Traffic Manager contains code from Sys::SysLog – a Perl library used as an interface to the UNIX syslog program. The following copyright notice applies:

Version: 0.27

License summary: Perl Artistic License

Copyright notice:

Copyright (C) 1990-2012 by Larry Wall and others.

This library is free software; you can redistribute it and/or modify it under the same terms as Perl itself. See the Perl Artistic License for details:

http://search.cpan.org/perldoc?perlartistic

CHAPTER 36 Index

Actions (Event Handling), 327

Active Connections page, 458

Active-active configuration, 40
  With loopback, 40

Active-standby configuration, 39, 41, 46

Activity monitoring. *See* Monitoring

Admin Server, 29, 351, 356, 357, 369
  Access Port, 351
  Audit Log, 372
  Control Port, 352
  Ports, 357

Alerts, 324

Application Firewall. *See* Virtual Web Application Firewall

Application server, 152

Architecture, 26, 52, 254, 464

Assigning requests, 40, 52, 56, 63, 80, 156

Authentication, 210, 213
  LDAP, 165
  SSL, 211

Authenticators. *See* Remote Authentication

Back end server, 31, 33, 37, 56, 80, 81, 197, 463, 468

Backing up, 90, 127, 128

Backup management, 126
  Partial backups, 128, 129
  Partial imports, 129, 392

Bandwidth management
  Adding a class to a pool, 263
  Adding a class to a virtual server, 263

Bandwidth Management
  Configuring, 262
  Example, 264
  overview, 261
  Using TrafficScript, 264

banner, 371

Buffering, 256

Catalogs, 53, 122
  CAs and CRLs, 211
  Monitors, 245, 248
  Rules, 64
  Service Protection, 65
  SSL Certificates, 65, 212, 215, 217

Certificate authority (CA), 210, 211, 215
  Importing, 218

Certificate revocation list (CRL), 211
  Importing, 218

Certificate signing request (CSR), 210

Client First. *See* Protocols, Generic Client First

Closing a connection, 205

Cloud Credentials, 94, 141

Cloud SteelHead. *See* SteelHead

Cloud Traffic Manager. *See* Traffic Manager Virtual Appliance

Cluster, 53
  Session persistence with a cluster, 199, 201, 202, 203
  Sizing the cluster, 33

Config Summary, 122

Configure script, 353, 464

Connection Activity report, 458

Connection Analytics, 69

Connection limiting, 66

Connection Limiting
Pool, 86
Connection Management
Pools, 85
Virtual servers, 72
Connection refused, 464, 465
Connections, 70
Debugging, 138
Content Caching, 28, 276
Configuring, 276
configuring web cache, 278
Disk-based Caching, 280
Monitoring, 280
Overview, 276
Policy, 281
TrafficScript, 283
Variants, 282
Content compression, 68, 133
Content management, 33
Control API, 374
C Sharp Example, 375
Mono Example, 375
overview, 374
Perl example, 374
Cookie settings, 72
Cookies, 203
Rules and cookies, 155, 161, 208, 459
Session persistence with …, 202
Session persistence with cookies, 83, 200, 201, 202, 203, 208
Creating a cluster, 56
Creation of a pool and a virtual server manually, 55
Current Activity graph, 133, 138
Customer prioritization rule, 161
Denial of Service, 65, 250, 253, 254, 255, 256, 257, 259
Distributed, 65, 253
Diagnosing faults, 27, 456, 462, 463, 465
Disabling nodes. *See* Pools
DNS, 26, 31, 402, 415
Listening on specific DNS addresses, 56
Monitoring activity, 133
Public addresses, 39, 40, 52, 464
Restricting access by DNS name, 357
Round-robin DNS, 40
Draining nodes. *See Pools*, *See Pools*, *See Pools*, *See* Pools
Error 400, 465
Error log. *See* Logs
Errors
Common configuration errors, 464
HTTP errors, 259
IP address allocation, 465
Evaluating the Traffic Manager, 31, 464
Event Handling, 324
Event Log, 121, 372, *See* Logs
Event Types, 325
Events
Custom, 137
Examples
TrafficScript, 152, 160
Universal session persistence, 208
External Program monitor, 248
Fail-over. *See* Fault tolerance
Failure pool. *See* Pools
Fastest Response Time load-balancing algorithm, 81
Fault tolerance, 31, 54
Back end, 89, 463
Front end, 33, 463, 464
Settings, 352
Fault Tolerance, 110
Broadcasts, 113
File system security, 354
Firewall

Configuration, 351
Firewalling, 29, 32, 254, 351, 352, 354, 357
techniques, 351
Front end server setup, 33
Front Page, 121
FTP. *See* Protocols, FTP
FTP (secure). *See* SSL-wrapped FTP
Generating test request, 462
Generic Client First. *See* Protocols, Generic Client First
Generic Server First. *See* Protocols, Generic Server First
Generic Server First banner, 177
Generic streaming. *See* Protocols, Generic streaming
Geographic load balancing. *See* Global Load Balancing
Getting help, 467
Global Load Balancing, 131, 398, 415
Global Server Load Balancing, 416, *See* Global Load Balancing
Global settings, 125
Google search request rule, 162
Hardware requirements, 26, 31, 34
Minimal install, 31
Headers
Content compression, 68
Filtering for service protection, 255, 259
Host header troubleshooting, 465
Logging values, 70
Rewriting, 152
With decrypted HTTPS, 220, 223
X-Cluster-Client-Ip, 72
Help pages, 158
Historical Activity graph, 137
Home Page, 121
HTTP. *See* Protocols, HTTP
HTTP attacks, 254
HTTP content caching, 67
HTTP headers within your error file, 78
HTTP monitor, 245, 248
HTTP redirects, 205
httpclient, 462
HTTPS, 465
Session persistence, 200, 201
Initial configuration, 52
Interface-to-Subnet mapping, 107, 109
Intranets, 211
IP addresses
Permanent, 31, 37
Traffic IP addresses, 27, 34, 37, 39, 40, 42, 48, 53, 105, 353, 463, 465
Configuration, 107
Passive, 109
Shared Traffic IP addresses, 107
Transfer, 114
Troubleshooting, 116
IP transparency, 35
cluster of traffic managers, 42
local routing problems, 37
routing configuration, 36
IP-Based session persistence, 199, 200, 203
IPv6
features in the Traffic Manager, 50
general features, 49
overview, 49
technical restrictions, 50
tuning Duplicate Address Detection on FreeBSD, Linux and Solaris, 50
Java
calling extensions from TrafficScript, 171
compiling a Java extension, 172

configuring the traffic manager, 172
introduction, 171
technical requirements, 172
Keepalives, 72, 85
Least Connections load-balancing algorithm, 81
Load balancing, 52, 80
Algorithms, 80, 461
And session persistence, 197, 205
Priority lists. *See* Priority lists
Location Header settings, 72
login failures
number permitted, 370, 371
Login to Admin Server, 54, 357, 370
Logs
Event Log, 125, 456, 463, 465
With service protection, 256
Requests, 70
Writing logs with TrafficScript, 156, 458
Loops, infinite. *See* Infinite loops
Low-level settings, 74
Managing a new service, 54
Map of the world, 138
Memory
Attacks, 253, 259
Settings, 73, 260
Monitor Application Cookies, 200, 202
Monitoring
Activity, 27, 92, 132
External Monitors, 250
Health, 53, 89, 245, 248, 249
SNMP, 133
writing in Perl, 251
Monitoring Health
advanced monitors, 246
basic monitors, 246
built-in monitors, 246
custom monitors, 248
overview, 244
per-node and pool-wide monitors, 249
protocol specific monitors, 246
Monitoring performance using SNMP, 133
Multiple-redundant configuration. See N+M redundancy
Multi-site cluster management, 393
N+M redundancy, 42
TrafficCluster, 42
Network layout, 26, 31, 34, 352
Networking
VLAN tagging, 336
NFS file server. *See* File system, shared
Nodes. *See* Pools
OCSP, 65, 211, 224
OCSP Stapling, 228
Online Certificate Status Protocol. *See* OCSP
Operating system
Security, 253, 351, 354
OSPF, 49
Passive Health Monitoring
Enabling and Disabling, 244
PEM-encoded files, 216, 218
Perceptive load-balancing algorithm, 81
Permissions groups. *See* User permissions
Ping monitor, 245, 248
PKCS compliant hardware, 233
Pools
Autoscaling, 141
Connection Management, 85
Creating, 55, 80
Default, 52, 54, 55, 56, 63, 156
Disabling nodes, 91

Draining nodes, 91, 92, 140, 141, 205, 206, 461
Editing, 80
Failure, 89, 90
Load balancing, 80
Monitoring health, 89, 244, 245, 249
Priority lists, 90, 91
session persistence, 198
Session persistence, 83, 154
SSL Encryption, 85, 228
POP3, 33, 52
Priority lists, 90, 91
Protocols, 52, 61
And session persistence, 199
Firewall considerations, 351
FTP, 183
Generic Client First, 176
Generic Server First, 175
Generic streaming, 178
HTTP, 178
RTSP, 187
SIP, 189
SMTP, 182
SSL. *See*
Random Node load-balancing algorithm, 81
Rate shaping
graphing, 270
web spiders, 269
Rate Shaping, 265
Configuring, 266
Specific, 268
TrafficScript, 268
Real Time Streaming Protocol. *See* Protocols, RTSP
Remote Authentication, 165
Request Rate Shaping. *See* Rate Shaping
REST API, 61
Round Robin load-balancing algorithm, 80, 461
Routing with rules, 64, 152, 160, 162, 208, 212
RTSP. *See* Protocols, RTSP
advantages of using the Traffic Manager, 188
related protocols, 188
usual operation, 187
RTSP custom monitor, 248
RTSP ports, 188
RuleBuilder, 155
Secure FTP. *See* SSL-wrapped FTP
Securing the traffic manager, 32, 254, 354, 356
Security
restricting access, 357
Server First. *See* Protocols, Generic Server First
Server First banner. *See* Generic Server First banner
Service Level Monitoring, 271
Adding a Class to the Catalog, 272
Applying a Class to a Virtual Server, 273
Configuring, 272
Examples, 273
Graphing, 275
prioritizing resources, 274
TrafficScript, 273
Service protection, 65, 66, 255
Access restrictions, 258
Basic settings, 256
Connection limiting, 255, 257
Creating a class, 256
HTTP-specific settings, 258
malformed HTTP filtering, 255
Monitoring, 133
Network Access Restrictions, 254
Rule-Based protection, 255

Rules, 154, 259
Testing a class, 256
Using a class, 259
Service Protection, 253
Enabling, 255
Performance, 260
Session Initiation Protocol. *See* Protocols, SIP
features, 189
monitor, 248
operation, 189
operation modes with the traffic manager, 192
Session persistence, 83
configuring, 198
Failure modes, 83
In each pool, 80
Load-balancing implications, 197
Methods, 199
Monitor Application Cookies, 202
Named Node, 199, 201
node failure options, 204
Overview, 197
SSL Session ID, 203
Transparent Session Affinity, 201
Universal, 152, 154, 201, 208
Using cookies, 83
X-Zeus-Backend cookie, 203
Session Persistence
ASP.net, 202
Sizing, 206
With ASP.net, 200
Session persistence mapping, 204
Settings
cookies, 72
Location Header, 72
Shared file system, 33, 35, 85, 249
Simple Network Management Protocol. *See* SNMP
SIP. *See* Protocols, SIP
SMTP, 33, 52, 463, *See* Protocols. SMTP
SNMP, 133
authentication and privacy, 136
Engine ID, 135
traps, 328
versions, 133
SOAP, 63, 162, 163
SSL. *See* Procotols, SSL
Catalogs, 212, 215
Certificate Sign Request (CSR), 215
Certificates, 27, 65, 210
Backing up, 128
Client, 213
Copying, 215
Creating, 213, 214
For Admin Server, 356
Importing, 216
Self-signed, 214, 215
Server, 213
Client certificates
With decryption, 65, 211, 220, 223
Configuring Certificates, 213
Decryption, 27, 61, 204, 212, 465
Configuring, 64, 219, 220, 223
Decryption Wizard, 212
Encryption, 27, 64, 80, 210, 212, 432
Configuring, 85, 228
HTTPS protocol, 210
intermediate certificates
adding, importing and updating, 217
overview, 210
Session ID cache, 227
SSL Session ID persistence, 83, 200, 203
ssl_enhance, 230
ssl_trust_magic, 230
SSL-wrapped FTP, 185
STARTTLS. *See* Protocols. SMTP
Static (web) content, 33, 83, 156
Steelhead

Discovery Agent, 348
SteelHead, 347
Integration, 347
Superuser privileges, 353, 354, 464
TCP, 83, 245, 352
TCP Connect monitor, 248
TCP Transaction monitor, 248
Technical support, 457
telnet, 462, 465
Testing individual nodes, 461
Testing tools and techniques, 456
Throughput, maximizing, 35
Timeout settings, 72, 73, 370
TLS
Server Name, 221
Traffic IP group. *See* IP addresses
Traffic IP Networks, 107, 109
Traffic Manager
Product versions, 23
Traffic Manager Virtual Appliance
Date And Time, 345
IP Forwarding, 342
NAT, 342
Network Configuration, 335
VLANs, 336
TrafficScript, 259
Applying rules, 63, 159
Authentication, 165
Converting from RuleBuilder, 157
Creating rules, 157
Documentation, 153, 158
Events, 137
Examples, 152, 160, 161, 162, 208
including extensions, 171
including Java in,, 171
Java support, 27, 171
overview, 151
Request rules, 64, 158, 159
Response rules, 158, 159
rulebuilder, 155
when to use rules, 154
Writing logs, 459
XML support, 162
Transparent Session Affinity, 200, 201
UDP, 72
SIP, 196
Universal session persistence, 154, 199, 201, 204, 208
Upgrading
Back ends, 91, 140
OS, 354
URL
Attacks, 255, 258, 259
Of Admin Server, 357
Redirection, 83, 205
Rewriting, 152
User permissions
In the OS, 353
In the traffic manager, 361, 362, 369, 370, 461
Users
Suspended, 370
Virtual servers, 61
Applying rules, 63, 159
Applying service protection, 65, 66, 259
Connection management, 72
Content compression, 68
Creating, 54, 56
Decrypting SSL traffic, 64, 219, 220, 223
Editing, 56, 61
Request logging, 70
Virtual Web Application Firewall, 23, 143
Viruses, 154, 212, 253
Viruses in DoS attacks, 253
VLAN (networking), 336

Web server, 31, 33, 39, 56, 68, 245, 254, 259, 465
Web services, 63, 162
Web worms, 154, 212, 253, 254, 259
Weighted Round Robin load-balancing algorithm, 80
Wizards, 54, 92, 141
  Disable a Node, 92
  Drain a Node, 92
X-Cluster-Client-Ip, 35, 72, 229
XML, 63, 162, 163
XPath, 63, 162, 163
X-Zeus-Backend cookie, 200, 203
`zconf`, 126, 390
  commands, 390
  Exporting backups, 391
  Listing config files, 392
  Partial Imports, 129, 392